UNIVERSIDAD AUTONOMA DE MADRID

ESCUELA POLITECNICA SUPERIOR

Grado en Ingeniería Informática

TRABAJO FIN DE GRADO

Adaptación de estímulos en interfaces cerebro-máquina que utilizan potenciales visuales evocados

Ana Aguilar Santamaría
Tutor: Pablo Varona Martínez

Mayo 2016

Adaptación de estímulos en interfaces cerebro-máquina que utilizan potenciales visuales evocados

AUTOR: Ana Aguilar Santamaría
TUTOR: Pablo Varona Martínez

Grupo de Neurocomputación Biológica (GNB)
Dpto. de Ingeniería Informática
Escuela Politécnica Superior
Universidad Autónoma de Madrid
Mayo de 2016

Resumen (castellano)

El gran progreso tecnológico en el ámbito biomédico y en el procesamiento de datos de las últimas décadas ha permitido el avance en los diversos métodos de interacción con el entorno y los dispositivos, surgida fruto de las necesidades de personas que padecen severas discapacidades motoras.

Una de las nuevas vías de comunicación con las máquinas con más auge actualmente en gran diversidad de disciplinas, entre las que no sólo destaca la clínica, sino también lacomunicación, el entretenimiento, la seguridad y la educación, son las interfaces cerebro-máquina (BCI), que se valen del análisis en tiempo real de la actividad eléctrica cerebral adquirida mediante registros neurofisiológicos con electroencefalografía (EEG).

El presente proyecto tiene como objetivo principal el desarrollo de una interfaz gráfica fácilmente configurable, adaptable al usuario y con aplicaciones reales, buscando la máxima fiabilidad en el control de la misma con la intención de minimizar y solventar las dificultades que se presentan en las personas con parálisis física parcial o total, valiéndose de estimulación visual para originar la actividad eléctrica cerebral deseada, prestando especial atención a la parametrización de dichos estímulos para la optimización individualizada en cada usuario. Esto se realizará mediante una adecuada presentación y selección de estímulos y parámetros involucrados en el interfaz gráfico.

Se crearán tres aplicaciones para la comunicación mediante deletreo, la comunicación mediante ideogramas y el control del movimiento de dispositivos; cuyas interfaces gráficas pueden ser fácilmente configurables antes de ser ejecutadas. Para ello, se hará uso de estimulación parpadeante que genera potenciales visuales evocados de estado estacionario (SSVEP) registrados mediante un casco de EEG de bajo coste disponible en el laboratorio del Grupo de Neurocomputación Biológica de la Universidad Autónoma de Madrid. Los experimentos de validación indican que la interfaz desarrollada puede adaptarse a distintos tipos de interacción.

Abstract (English)

The large technological development in the biomedical and data processing domains in the last few decades has led to the advance in the ways in which a person interacts with the environment and technological devices. This is a direct consequence of addressing the needs of people with severe motor disabilities.

Nowadays, one of the most popular methods to communicate with machines involving many different context, such as medical, communication, entertainment, education and security applications are brain computer interfaces (BCI). A BCI process and analyzes the electrical activity of the brain obtained by electroencephalography (EEG).

The main objective of the project is to develop an easily configurable BCI graphical interface adapted to the user. In order to minimize and solve problems in people with partial or total paralysis, this project looks for the most reliable control and the best adaptation to the user in brain computer interfaces, using visual stimulation to generate the desired electrical brain activity. Special attention will be paid to stimulus parametrization to individually optimize it for each user by properly choosing and presenting stimuli and personalizing the parameters related to the graphical interface.

Three applications (communication by spelling, communication using ideograms and device movement control) are validated measuring Steady Stationary Visual Evoked Potentials (SSVEP) generated by flickering stimulation and using a low-cost EEG system available at Universidad Autónoma de Madrid's Biological Neurocomputation Group lab. These applications offer new ways of communication and device control to people with motion disorders. Additionally, the user will be able to easily set up the graphical interface parameters. The validation tests point out that the developed graphical interface adapts to many different kinds of interactions.

Palabras clave (castellano)

Interfaz cerebro-máquina (BCI), electroencefalografía/electroencefalograma (EEG), Potenciales Visuales Evocados de Estado Continuo (SSVEP), discapacidad motora severa, comunicación, usabilidad, accesibilidad.

Keywords (inglés)

Brain Computer Interface (BCI), electroencephalography/electroencephalogram (EEG), Steady State Visually Evoked Potentials (SSVEP), severe motor sensory disorder, communication, usability, accessibility.

Agradecimientos

Me gustaría expresar mi agradecimiento a todos mis compañeros de carrera que han formado parte de mi viaje a través de la vida universitaria y la ingeniería informática, en especial a Sergio por haberme acompañado incondicionalmente estos años, haberme apoyado en todas mis decisiones y haber dado el mejor giro de trescientos sesenta grados a mi vida.

Especial agradecimiento a mi tutor, Pablo. Gracias por haberme sugerido un tema tan interesante y su implicación en el proyecto. Así como a todas las personas que de una forma u otra han participado en este trabajo, en especial a Álvaro Morán, por su aporte fundamental para la resolución de éste.

Gracias a Sergio, Elena, Víctor y Angeli, entre otros muchos compañeros de laboratorio, por hacer de las inagotables horas de prácticas y las comidas en la cafetería de la escuela momentos inolvidables de risas.

A mis amigos y mis padres que me han acompañado todos estos años, aconsejando y sintiendo también en parte suya mi experiencia en esta carrera, gracias por todo.

INDICE DE CONTENIDOS

INDICE DE FIGURAS

INDICE DE TABLAS

1 Introducción

1.1 Motivación

El desarrollo de la tecnología de interfaces hombre-máquina ha experimentado un gran crecimiento en los últimos años. Pese a que el uso de los periféricos más básicos de interacción con las máquinas, como el ratón o el teclado, siguen teniendo un uso habitual en la vida diaria, la aparición de otros métodos, como el control de dispositivos mediante órdenes por voz, gestos o interacción ocular, entre otros, ha supuesto una significativa mejora en la usabilidad de los sistemas promoviendo una interacción más natural e intuitiva. Estos avances han sido posibles gracias al gran progreso en los sistemas de sensorización y el aumento en la capacidad de procesamiento de señales en tiempo real.

Como vertiente de las interfaces hombre-máquina, surgen las interfaces cerebro-maquina (BCI), fruto de la necesidad de mejora de, no solo la usabilidad, sino también la accesibilidad a estos dispositivos para personas que no disponen de todas sus facultades motoras o bien no pueden hacer uso de ellas en un determinado momento. Se basan en el control de dispositivos mediante la captura y análisis de la actividad cerebral. Las principales funciones de estos sistemas son permitir al usuario expresar sus ideas e intenciones, o evaluar su estado. En algunas aplicaciones, los usuarios objetivo no tienen por qué padecer discapacidades motoras, ya que su uso se enfoca como herramienta de medida psicológica que aporta información emocional (Abdulkader et al., 2015).

En muchos casos estos interfaces pretenden mejorar la calidad de vida de aquellos usuarios que padecen discapacidades físicas graves aunque su nivel cognitivo es alto (Pultarova, 2014). Una BCI proporciona a personas que sufren trastornos tales como esclerosis lateral amiotrófica (ELA), derrames, lesiones cerebrales o en la médula espinal, entre otros (Cipresso et al., 2012) nuevas vías de comunicación e interacción con su entorno. Debido a esto, se ahondará en las posibilidades que ofrecen los interfaces mediante el uso de métodos no invasivos, que brindan mayor comodidad y usabilidad al usuario aunque conllevan una serie de limitaciones en la magnitud de la señal cerebral obtenida. El método no invasivo más utilizado en este tipo de experimentos es la electroencefalografía (EEG) que detecta la actividad eléctrica cerebral. Los métodos invasivos, como la electrocoticografía, sin embargo, tienen un componente de riesgo mayor debido a la necesidad de pasar por quirófano, por lo que únicamente se lleva a cabo en casos de necesidad extrema, pese a que la calidad de la señal capturada es más alta (Leuthardt et al., 2006).

En las BCI más eficaces, los usuarios son sometidos a una serie de estímulos para evocar actividad cerebral-estereotipada que se puede relacionar con la voluntad del usuario para realizar una acción. La respuesta cerebral producida bajo la influencia de estímulos sensoriales se traduce en potenciales que pueden ser analizados y cuantificados. Los estímulos sensoriales más comunes en este tipo de interfaces son los estímulos visuales y auditivos. Otros como la visualización motora también se pueden usar en otro tipo de fines como la rehabilitación locomotriz de extremidades no funcionales mediante el control de electrodos colocados sobre ellas, dado que esta estimulación activa las mismas áreas del cerebro que se activan con el movimiento real (Mulder, 2007). El fenómeno neurológico empleado este proyecto son los potenciales visuales evocados de estado estacionario (SSVEP, siglas en inglés), señales cerebrales generadas en respuesta a una estimulación visual parpadeante, cuya frecuencia se corresponde con la de los estímulos, ya que los

potenciales obtenidos son de igual valor a la frecuencia de parpadeo del estímulo o múltiplos de ella (armónicos) (Herrmann, 2001; Kuś et al., 2013).

Las interfaces cerebro-máquina pretenden obtener la máxima fiabilidad y precisión posible en la identificación de comandos, limitando la cantidad de errores en las detecciones, primordialmente en aquellos casos cuya aplicación esté orientada a la ayuda a personas discapacitadas. El método a seguir en este proyecto es la adaptación al usuario con el establecimiento de los diversos parámetros de manejo de los estímulos mediante el diseño de una interfaz flexible y adaptable. En concreto en este trabajo se empleará un interfaz configurable de estímulos parpadeantes generados en la pantalla de un ordenador.

1.2 Objetivos

El objetivo esencial de este proyecto es el desarrollo de la interfaz de usuario altamente flexible para un sistema cerebro-máquina basado en estímulos visuales generados en una pantalla que evocan SSVEPs registrables mediante electroencefalograma. Esta interfaz brindará al usuario la mejor experiencia posible, obteniendo respuestas fiables y en tiempo real. Para ello, es necesaria una adaptación al tipo de interfaz y al usuario, por lo que será fácilmente reconfigurable.

Esta adaptación conllevará diversos estudios para realizar una selección automática y personalizada de los mejores parámetros que influyen en la creación de los estímulos evocados por la interfaz, entre ellos el color, la forma, el tamaño, etc.; tomando, en consideración que serán necesarias tantas frecuencias distintas como opciones posibles se ofrezcan al usuario. La elección de dichas frecuencias óptimas se realizará mediante la ayuda de un algoritmo de ciclo cerrado, para lo cual ha de repararse en las limitaciones que la propia frecuencia de la pantalla impone sobre esta optimización.

1.3 Organización de la memoria

Dichos objetivos se contemplarán desde diversos puntos de vista en los que se organiza esta memoria que consta de los siguientes capítulos:

- El **estado del arte** expondrá el contexto histórico y tecnológico que ha llevado a la creación de las interfaces cerebro-máquina, especialmente atendiendo al contexto de estimulación parpadeante en pantallas, las diversas técnicas y parámetros a considerar en su desarrollo y las características de los SSVEPs.
- En el **diseño** de la interfaz gráfica de estimulación y control de comandos se realizará una toma de decisiones argumentadas, centrándose en la usabilidad y fiabilidad de la aplicación, estudiando el uso de qué tecnologías son las más convenientes para este proyecto.
- El **desarrollo** de la memoria expondrá los métodos de implementación del diseño establecido en el capítulo anterior elegidos, documentando la funcionalidad de todos los módulos desarrollados de la interfaz de usuario.
- Se hará especial énfasis en la **integración** de la interfaz desarrollada con las demás etapas para la creación completa de una interfaz cerebro-máquina, aportadas por el Grupo de Neurocomputación Biológica de la Universidad Autónoma de Madrid, necesarias para la realización de **pruebas** de validación y obtención de **resultados** para la comprobación de la adaptación al usuario.
- Para concluir se aportará una serie de **conclusiones, generalización del trabajo desarrollado** y los puntos a abordar en un **trabajo futuro** para la mejora y extensión del ámbito de aplicación de este proyecto.

2 Estado del arte

2.1 Interfaces de comunicación entre usuario y máquina

2.1.1 Interfaces hombre-máquina

Las interfaces hombre-máquina, *Human Machine Interface* (HMI), también conocidas como interfaces de usuario, tienen como finalidad gestionar la interacción entre los usuarios y los dispositivos electrónicos. Se encuentran en constante cambio y afán de mejora, persiguiendo mayor accesibilidad y usabilidad para el usuario (Dix, 2009) (Lebedev, 2014).

Con objetivo de llevar a cabo dicha interacción, estas interfaces, deben albergar la capacidad necesaria para la obtención y procesado de la señal originada en el usuario, así como de facilitar retroalimentación (*feedback*) en tiempo real al usuario.

Una vez superadas las limitaciones de procesamiento de los primeros sistemas y dispositivos, centrados en el uso de interfaces basadas en texto haciendo uso de interfaces de entrada como el teclado y el ratón, se impulsó la evolución de las HMI a otros tipos de interfaces de usuario dotadas de mayor accesibilidad gracias el control por voz, la interacción táctil o gestual o la interacción a través del seguimiento ocular, así como las interfaces cerebro-máquina, las cuales serán el centro de nuestro estudio.

2.1.2 Interfaces cerebro-máquina

La principal característica diferenciadora de las interfaces cerebro-máquina, *Brain Machine Interface* (BMI), también conocidas como interfaces cerebro-computador, *Brain Computer Interface* (BCI), con resto de interfaces hombre-máquina mencionadas anteriormente, reside en que la interacción entre usuario y máquina debe producirse sin precisar de actividades motrices (Graimann et al., 2010).

Las BCI ven sus inicios a principios del siglo XX, cuando Hans Berger, neurólogo alemán, continuando con los estudios acerca de los impulsos eléctricos del cerebro de Richard Caton, en 1924, obtiene el primer registro de la actividad cerebral de un cerebro humano mediante una encefalografía. Sin embargo, no fue hasta la década de los 70, cuando se evidencia que determinados potenciales cerebrales se correlacionan con movimientos reales y determinadas actividades mentales. A finales de los 90 quedó patente el potencial médico de las BCI mediante la implantación de un electrodo en el córtex motor de un paciente con parálisis de cuello hacia abajo que había perdido la facultad del habla, gracias al cual fue capaz de comunicarse controlando un cursor en una pantalla bidimensional (Galindo Merchán, 2008).

Su nacimiento surge debido a la necesidad establecer nuevos medios de comunicación con dispositivos electrónicos para personas con graves discapacidades motoras, no obstante, pueden resultar también útiles en el estudio de las propias enfermedades, mediante el uso de estímulos y el entrenamiento de las interfaces, analizando la evolución del usuario en su control.

Pese a que cada grupo de investigación especializado en este ámbito de las comunicaciones hombre-máquina ha desarrollado su propio sistema BCI, todos ellos comparten un esquema de funcionamiento similar (Galindo Merchán, 2008):

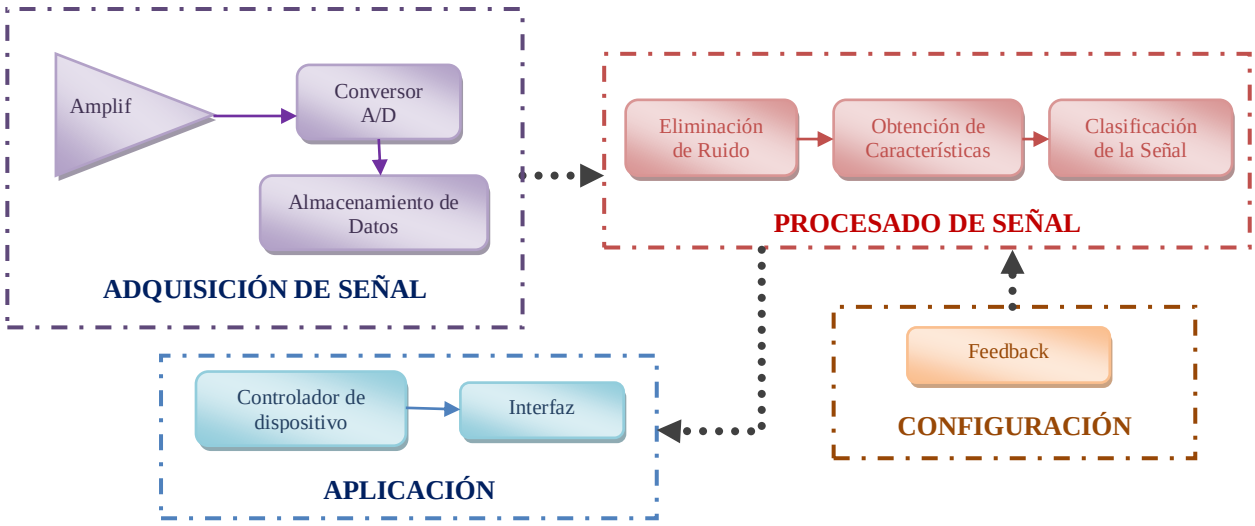


Figura 2-1: Esquema de interfaces cerebro-máquina
-adaptada de (Galindo Merchán, 2008)-

En él distinguimos cuatro fases principales:

En primer lugar se realiza la **adquisición de señal**, mediante la cual se realiza el registro de la actividad cerebral del usuario. Se capturan las señales neurológicas con la ayuda de sensores, que han de ser amplificadas y digitalizadas antes de su posterior procesado. Pese a que el procesamiento de las señales en tiempo real no requiere del almacenamiento de la señal digitalizada, casi todos los sistemas BCI realizan esta etapa con el fin de posibilitar el análisis y procesado posterior de la misma.

Una vez digitalizada la señal, se procesa, en tiempo real, realizando el acondicionamiento y la **eliminación de ruido** que la contamina, resultante de otros tipos de actividad bioeléctrica, como pueden ser los movimientos oculares o musculares, entre otros, y de interferencias electromagnéticas. Se analizan las **características más representativas** de la señal en relación con el fenómeno neurológico que estemos analizando, lo que permite **clasificar la señal** dentro de alguno de los conjuntos formados por los comandos que la interfaz puede ejecutar (decodificación).

La **aplicación** recibe los comandos de control y es la encargada de realizar las acciones correspondientes a través del controlador del dispositivo. Además, durante esta etapa se puede incorporar *feedback* al usuario.

Por último, la **configuración** permite el ajuste de los parámetros del sistema en función de los resultados obtenidos y la retroalimentación con el usuario. Este *feedback* puede producirse de distintas formas dependiendo de la finalidad de la interfaz. Así como si se pretende el control de movimiento de un dispositivo externo, la retroalimentación vendrá dada por el propio movimiento del mismo, en otro tipo de aplicaciones es posible que sea necesaria un *feedback* adicional, por ejemplo una indicación visual por pantalla al usuario.

2.2 Señales neurológicas

2.2.1 Origen de las señales

Un aspecto importante a tener en cuenta en el diseño de las interfaces cerebro-máquina es su clasificación según la naturaleza de las señales registradas.

Las **señales endógenas** o **no evocadas** son generadas en el interior del cerebro sin requerir ningún tipo de estimulación externa. Se trata de señales muy dependientes del estado de emocional y físico (dolor, cansancio…) del usuario. Los sistemas BCI endógenos dependen de la habilidad del usuario para controlar su actividad electrofisiológica y requieren un periodo de entrenamiento intensivo. Dos ejemplos de los sistemas endógenos más utilizados son (Hornero et al., 2012):

- Las BCI basadas en **potenciales corticales lentos**, *Slow Cortical Potencials* (**SCP**), se corresponden con modificaciones lentas de voltaje generadas sobre el córtex cerebral de duración variable entre 0,5 y 10 segundos. Los niveles negativos de SCP se suelen asociar al movimiento y otros cambios en la actividad cortical. Por ende, niveles positivos de SCP coinciden con una disminución de la actividad cortical. Se ha demostrado que el control de estos potenciales puede aprenderse (Birbaumer, 1999).
- Las BCI basadas en **imágenes motoras o ritmos sensoriomotores** se apoyan en los cambios de amplitud en los ritmos sensoriomotores producidos por dos o más clases de imágenes motoras como el movimiento de extremidades o tareas mentales tales como los cálculos aritméticos, la rotación de un cubo, etc. Estos ritmos albergan dos tipos de variaciones en amplitud Los ritmos sensoriomotores comprenden, además, dos tipos de variaciones en amplitud: **ERD** (*Event-Related Desynchronization*), que implica una atenuación de la amplitud de los ritmos, y **ERS** (*Event-Related Synchronization*), que implica un aumento de la amplitud de los mismos. (Neuper et al., 2006)

Las **señales exógenas** o **evocadas** son aquellas generadas mediante una estimulación externa. Dado que la actividad electrofisiológica es evocada por estímulos externos, el entrenamiento no es estrictamente necesario en los sistemas BCI exógenos. Dos ejemplos de los sistemas exógenos más utilizados son:

- Las BCI **basadas en potenciales evocados P300**, los cuales representan picos de amplitud positivos detectados en un encefalograma en torno a 300 milisegundos después de que el usuario perciba un estímulo infrecuente auditivo, visual o somatosensorial. Se presentan mezclados con otros estímulos más comunes, lo que permite que esta estimulación poco frecuente provoque la aparición de potenciales P300 en la actividad cerebral del usuario. Estos potenciales se observan en las zonas central y parietal del córtex cerebral (Nicolas-Alonso and Gomez-Gil, 2012).
- Las BCI **basadas en potenciales visuales evocado**s, *Visual Evoked Potentials* (*VEP*), los cuales son detectados en el electroencefalograma (EEG) en la corteza visual del cerebro, es decir, en la zona occipital del córtex. En función de la frecuencia de parpadeo de las imágenes evocadas, estos sistemas BCI pueden clasificarse como **transitorios**, *Transient VEP* (**TVEP**) si es menor a 6Hz, o de **estado continuo**, *Steady State VEP* (**SSVEP**), en caso contrario (Cui et al., 2004).

Un resumen de las características de la clasificación de las señales y los sistemas BCI según su origen se refleja en la Tabla 2-1, adaptada de (Nicolas-Alonso and Gomez-Gil, 2012):

Origen	Señales neurológicas	Ventajas	Desventajas
BCI Endógenas	SCP Ritmos sensoriomotores	Independiente de cualquier estimulación externa, por lo que resulta útil para personas con órganos afectados sensorialmente Depende de únicamente de la voluntad del usuario Útiles para el control de aplicaciones con cursor	Requieren un tiempo de entrenamiento (semanas-meses) No todos los usuarios son capaces de obtener el control de las señales Baja tasa de transferencia Necesidad de utilizar varios canales de EEG
BCI Exógenas	P300 VEP	Entrenamiento mínimo Configuración de señal de control sencilla y rápida Alta tasa de transferencia Únicamente se requiere un canal para el EEG	Atención permanente a estímulos externos, lo que puede producir cansancio en algunos usuarios.

Tabla 2-1: Ventajas y desventajas de las señales neurológicas según su origen

2.2.2 Métodos de adquisición de señales

Un factor relevante directamente relacionado con la resolución de las señales neurológicas obtenidas es el tipo de sensores que se utilizan para la adquisición de las mismas dependiendo de su objetivo: precisión o reducción del impacto en el usuario. Pueden distinguirse dos clases de métodos:

Los **métodos invasivos** tienen como objetivo la obtención de las señales más precisas posibles, reduciendo al mínimo el ruido, generalmente, accediendo directamente a las zonas del cerebro de donde se pretende adquirir las señales buscadas. Para logarlo, la técnica más usual en estos casos es la electrocorticografía (ECoG), tratándose ésta del registro de actividad eléctrica cerebral colocando electrodos en las zonas del córtex cerebral que se quieran estudiar tras realizar una craneotomía; de manera que se eliminan las interferencias que puedan producirse por el cráneo y cuero cabelludo que estarán presentes en los métodos no invasivos (Leuthardt et al., 2006).

En este tipo de intervenciones, suele emplearse la matriz de electrodos Utah, *The Utah Intracortical Electrode Array* (UIEA), una estructura de microelectrodos de silicona e iridio en forma de alfileres que se introducen a 1,5 milímetros en el córtex.

Los riesgos para el usuario que conlleva llevar a cabo estos métodos son evidentes, además de por la necesidad de someterse a una operación quirúrgica, por los posibles problemas de rechazo de su organismo que puedan surgir. Por este motivo, este tipo de métodos se suele reservar a personas con discapacidades de movilidad motora muy graves, personas ciegas como tentativa de recuperar la visión (Prieto et al., 2013) o personas que sufren de epilepsia con el propósito de encontrar el foco de producción de los ataques.

Los **métodos no invasivos**, por el contrario, se centran en reducir el impacto que estos sistemas puedan tener sobre los usuarios, realizando la adquisición de señales mediante electrodos externos colocados en la superficie externa del cráneo (Prieto et al., 2013). El inconveniente del uso de estos métodos reside en, como comentábamos anteriormente, el ruido que pueda producirse, así como el empobrecimiento del nivel de señal obtenido, que es del orden de microvoltios.

En esta categoría no invasiva de métodos, se mide la actividad eléctrica cerebral mediante electrodos colocados en el cuero cabelludo, amplificando y digitalizando la señal tal y como se analizó en la sección 2.1.2. Los electrodos se colocan en una superficie bastante amplia del cráneo y la resolución espacial obtenida no es muy elevada.

Algunos ejemplos de instrumentos no invasivos para la adquisición de señales neurológicas se ven representados en la Figura 2-2, imágenes publicitarias del casco Emotic Epoc (Emotiv, 2016) y g.Nautilus (g.tec Medical Engineering, 2016):

Figura 2-2: Cascos/gorros no invasivos para la obtención de señales

Otras técnicas de neuroimagen, aunque menos utilizadas debido a la necesidad de realizar instalaciones y disponer de equipos de alto coste, pueden ser:

La magnetoencefalografía (MEG), consistente en el registro de la actividad magnética cerebral mediante la detección de campos magnéticos creados mediante las corrientes entre las dendritas de las neuronas. La ventaja de esta técnica es la disminución de las distorsiones con respecto a la EEG, sin embargo requiere de máquinas voluminosas y con un coste muy elevado.

Las imágenes de resonancia magnética funcional, *functional Magnetic Resonance Imaging* (fMRI), detectan cambios en el volumen de sangre local cerebral, en los flujos sanguíneos cerebrales y los niveles de oxigenación durante la activación neuronal mediante campos electromagnéticos (Nicolas-Alonso and Gomez-Gil, 2012). La principal ventaja de esta técnica reside en su gran resolución espacial, debida a su capacidad de identificar con hasta 1 a 3 milímetros de resolución la zona del córtex con mayor actividad cerebral conforme a los niveles de oxígeno que haya en sangre. Sin embargo, como contra, ofrece menos resolución temporal que la EEG para adquirir las imágenes, aproximadamente cada 5 u 8 segundos (Monge, 2015) y solamente puede realizarse con maquinaria disponible en hospitales o centros de investigación.

2.3 SSVEP, Steady State Visual Evoked Potentials

En esta sección, se expondrá una explicación concisa del método utilizado en este proyecto basado en potenciales visuales evocados de estado estacionario (SSVEP) con el objetivo de entenderlo en mayor profundidad

La estimulación basada en SSVEP, como se explicó en la sección 2.2.1, trata con potenciales visuales evocados centrándose en la frecuencia de parpadeo de las imágenes. Cuando una persona focaliza su atención visual en uno de los estímulos que se le muestra continuamente parpadeando a una frecuencia igual o superior a 6Hz, se produce una resonancia en la zona visual del cerebro, la zona occipital, generando señales neurológicas a la misma frecuencia a la que se sometió el estímulo, como se puede apreciar en el esquema de la Figura 2-3, o múltiplos de ella, también llamados armónicos (Herrmann, 2001; Mora-Cortes et al., 2014). La diferenciación de las intenciones del usuario no resulta

una tarea trivial, puesto que debe tenerse en cuenta la dificultad que reside en el reconocimiento de los patrones concretos de respuesta esperados en la actividad cerebral recogida. Sin embargo, la evocación de actividad estereotipada con las señales parpadeantes, como la utilizada en este proyecto, consigue que el reconocimiento de la intención del usuario sea más sencillo. En los interfaces basados en SSVEP se presentan a la vez varios estímulos parpadeantes a distintas frecuencias que codifican posibles intenciones del usuario, como puede observarse en el panel de la derecha de la Figura 2-3.

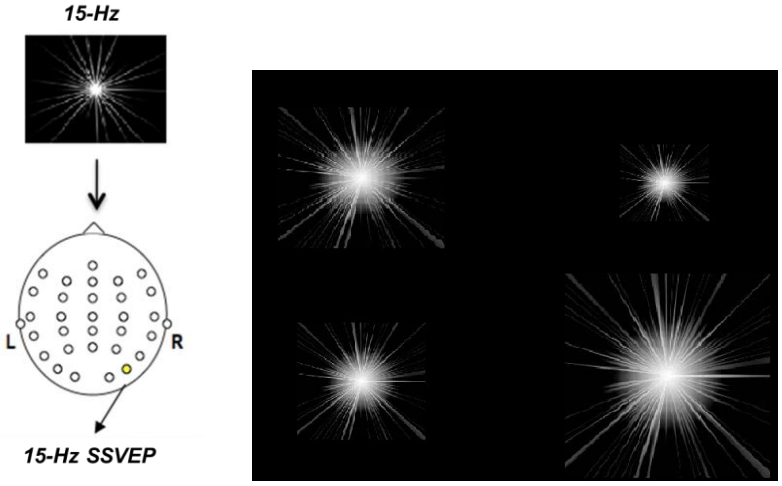


Figura 2-3: Ejemplo de estímulo y resonancia producida en SSVEP

2.3.1 Parámetros de estímulos SSVEP

Estudios en el ámbito de los potenciales SSVEP, distinguen tres **tipos de estímulos** posibles (Zhu et al., 2010):

Los **estímulos de luz** son aquellos que producen el estímulo mediante el parpadeo de LEDs, luces fluorescentes, luces de neón, etc., controlados por circuitos electrónicos integrados que regulan la intensidad lumínica y la onda de los estímulos.

Los **estímulos de gráfico único** se representan en una pantalla apareciendo y desapareciendo del fondo de la misma con una frecuencia dada. La imagen parpadeante se trata de un único patrón, como puede verse en la Figura 2-4, adaptada de (Zhu et al., 2010):

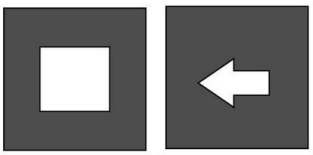


Figura 2-4: Ejemplos de patrones de gráfico único

Por el contrario, los **patrones invertidos** se basan en el parpadeo en una pantalla de dos patrones distintos, usualmente representados en blanco y negro en forma de tablero de ajedrez o líneas alternadas como puede apreciarse en la Figura 2-5, adaptada de (Zhu et al., 2010):

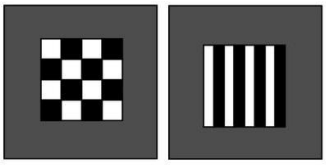


Figura 2-5: Ejemplos de patrones invertidos

Otros factores a tener en cuenta, en el caso del uso de estímulos representados mediante el parpadeo de imágenes en una pantalla, son la frecuencia, el color, el tamaño, el uso de un punto de referencia, la forma, y la distancia entre estímulos (Resalat et al., 2012). Los interfaces modernos basados en SSVEP o en combinaciones de SSVEP y otras modalidades utilizan optimizaciones dinámicas (Chang et al., 2016; Yin et al., 2015).

Las **frecuencias** utilizadas en SSVEP se pueden clasificar en tres bandas: banda baja, entre 1 y 12 Hz; banda media, entre 12 y 30 Hz; y banda alta, entre 30 y 60 Hz. En la mayor parte de experimentos con SSVEP se utilizan las bandas baja y media, ya que son aquellas que obtienen mejores resultados. Sin embargo, consideraciones subjetivas de sujetos sometidos a pruebas con dichas bandas las consideran molestas a la vista, por lo que puede aparecer fatiga visual fácilmente (Zhu et al., 2010). Por otro lado, hay evidencias de que existen frecuencias óptimas de estimulación para cada sujeto (Fernandez-Vargas et al., 2013). El **color** de los estímulos es un parámetro cuya eficacia es difícil de medir. Esto se debe a que los colores dependen de variedad de factores como el brillo, la saturación, el contraste, etc., así como su dependencia con las frecuencias utilizadas. Se han realizado diversos estudios con respecto a la influencia del color en las respuestas SSVEP (Duszyk et al., 2014; Singla et al., 2013). Algunos muestran que algunos colores son más dependientes que otros a la frecuencia utilizada; otros revelan que los colores afectan de modos distintos al segundo y cuarto armónico de los potenciales SSVEP obtenidos. Sin embargo, los factores que parecen más significativos en la obtención de potenciales con mayor amplitud son el brillo y contraste del estímulo con el color de fondo. En la Figura 2-6 (a), adaptada de (Duszyk et al., 2014), se puede ver que los colores con más contraste con el fondo negro, como son el blanco y el amarillo obtienen mejores resultados, mientras los colores con menor brillo como el azul o verde oscuros, recogen peores resultados.

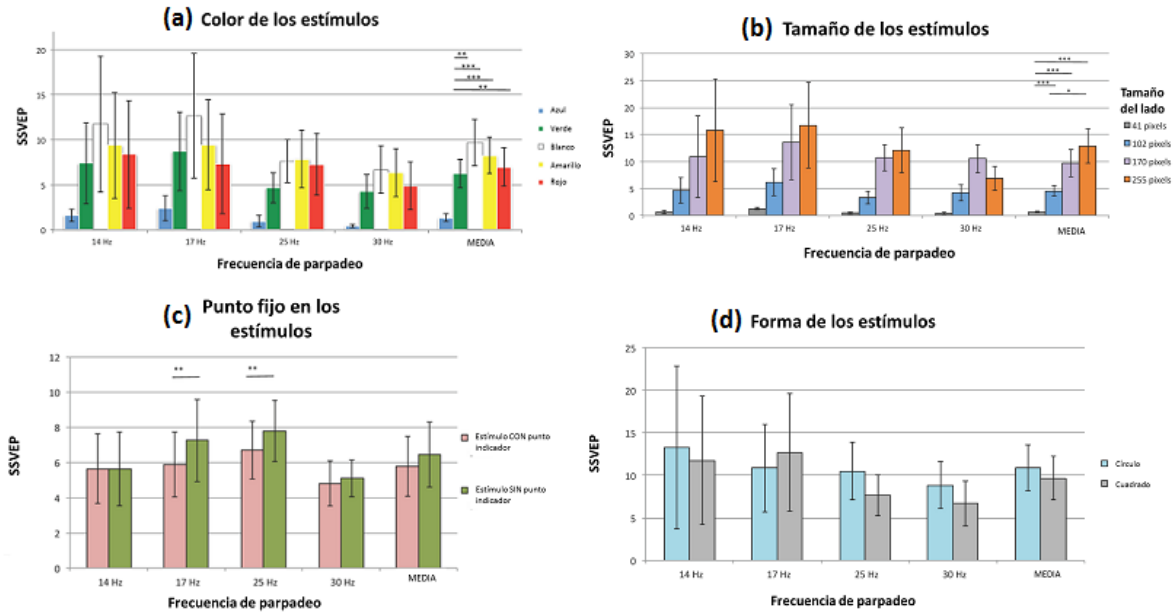


Figura 2-6: Gráfico de la relación entre la respuesta SSVEP y los parámetros del estímulo

Se ha demostrado empíricamente que el **tamaño** de los estímulos es directamente proporcional a la calidad de respuesta SSVEP; cuanto mayor es el tamaño, mayor es la amplitud de onda de la respuesta (Duszyk et al., 2014). Los autores de estos experimentos lo justifican asumiendo que un estímulo mayor activará áreas más extensas del córtex que

un estímulo pequeño. En la figura 2-6 (b), adaptada de (Duszyk et al., 2014), se pueden observar los resultados de dicho experimento y la clara relación entre los dos fenómenos:

Debido al cansancio que puede provocar en el usuario someterse a la estimulación visual en bandas de frecuencia bajas y medias en SSVEP, varias investigaciones han tomado en consideración la posibilidad de añadir un **punto de referencia**, indicador del estímulo en el que el usuario debe centrar su atención. Sin embargo, los resultados, representados en la Figura 2-6 (c), adaptada de (Duszyk et al., 2014), indican que la presencia de un punto de referencia baja, en la mayoría de casos, aunque no de forma significativa, la calidad de la respuesta SSVEP. Esto se debe a que al fijar nuestra atención en el punto de referencia y no en el propio estímulo parpadeante, la sensibilidad a la frecuencia del parpadeo baja. Sin embargo en media, las diferencias no son relevantes.

Se ha demostrado que la **forma** de los estímulos y la **distancia** entre los mismos no afecta de forma significativa a la magnitud de la respuesta SSVEP, como puede observarse en la Figura 2-6 (d), adaptada de (Duszyk et al., 2014).

En el caso de la distancia entre estímulos, se sospechaba que el parpadeo de los demás estímulos representados en la pantalla que se encuentran en el campo visual del usuario podrían interferir en la respuesta SSVEP del estímulo en el que el usuario centra su atención. Las pruebas demuestran que las distancias medias y grandes obtienen generalmente mejores resultados que las pequeñas, aunque no tienen un impacto de gran relevancia sobre la respuesta esperada.

3 Diseño

3.1 Primeras consideraciones

El objeto del presente proyecto consiste en la elaboración de una interfaz cerebro-máquina reconfigurable en cuanto a funcionalidad cuyo propósito reside en alcanzar una usabilidad máxima de la misma y la mejor adaptación posible al usuario, obteniendo siempre resultados fiables. La interfaz tendrá como objetivo determinar la intención del usuario dado un conjunto de acciones posibles.

En el ámbito de este proyecto, la **electroencefalografía** (EEG) supone la mejor elección como método de toma de señales debido a su bajo coste y la calidad de las mismas, siendo ésta la más eficaz con respecto a la resolución temporal y suficientemente fiable en sus resultados para el caso que nos ocupa: distinguir una elección dentro de un conjunto limitado y reducido de opciones en tiempo real. Otro dato a tener en cuenta para seleccionar el método de adquisición de señal que mejor se adapta a este proyecto es que las ventajas que nos proporcionan otros métodos no invasivos, mencionados en la sección 2.2.2, como puede ser la obtención de la señal con gran resolución espacial, resultan inviables para este proyecto.

Dentro de las múltiples alternativas para la toma de datos cerebrales, se ha escogido un casco cuyo montaje sea lo menos arduo posible con respecto a la colocación de los electrodos, pensando siempre la usabilidad para el usuario, una de las bases del proyecto. Se ha escogido un casco de electrodos que no requieren gel conductor debido sus ventajas en reutilización y a que pueden utilizarse durante periodos largos sin riesgos (Baba and Burke, 2008).

En relación al tipo de señales que se busca obtener, son óptimas aquellas que requieran de un entrenamiento mínimo o nulo debido a la necesidad de responder a las necesidades e intenciones del usuario en tiempo real, por lo que las señales de origen endógeno como SCP y ritmos sensoriomotores quedan descartadas. Además, se debe tomar en consideración que su eficacia depende del estado de ánimo del usuario y esto puede afectar negativamente los resultados obtenidos. Este argumento reduce el ámbito de señales que se quieren adquirir a las señales de origen exógeno o potenciales evocados. Puesto que los potenciales evocados P300 detentan una latencia de 300 milisegundos para la obtención de la señal, los potenciales evocados **SSVEP** constituyen la mejor alternativa.

Dado que las señales escogidas son potenciales evocados, se deberá generar una serie de estímulos al usuario, en este caso se utilizarán **estímulos visuales**. Para ello se hará uso de una pantalla con refresco de 60Hz mediante la cual se crearán los **estímulos de patrón único** sin punto de referencia, cuya frecuencia de parpadeo se verá limitada por la frecuencia de refresco de la pantalla. Idealmente, para garantizar los mejores resultados, se deberían utilizar frecuencias inferiores a la mitad de la frecuencia de refresco y sus subarmónicos, sin embargo, en este proyecto se emplearán también frecuencias no divisoras de la frecuencia de refresco con el objetivo de medir su efectividad. Se ha escogido este tipo de estímulos y no otros, como los estímulos de luces LED, por su flexibilidad en cuanto a la configuración espacial del interfaz. Asimismo, los estímulos de luces requieren un uso hardware específico, lo cual añade mayor complejidad al sistema y lo hace menos genérico.

Un estudio reciente realizado por los departamentos de Neurociencia e Ingeniería Biomédica de la Universidad de California, la Academia de Ciencias China y la

Universidad Tsinghua en Beijing, evidencia que los potenciales **SSVEP** son la **opción más rápida y eficaz** utilizando **estímulos presentados por pantalla y haciendo uso de la información de fase**. Con esta estrategia han conseguido crear una interfaz cerebro-máquina que permite deletrear 60 caracteres por minuto mediante la modulación de la unión de la frecuencia y la fase de la señal capturada (Chen et al., 2015).

3.2 Esquema de funcionamiento

Este proyecto sigue el diseño de los sistemas BCI representado en la sección 2.1.2, centrándose en la interfaz gráfica y las múltiples aplicaciones de los sistemas BCI para mejorar la accesibilidad y comunicación en personas con graves discapacidades motoras. Con el fin de acreditar la utilidad de estas aplicaciones, el proyecto se apoyará en las etapas de captura, procesado de datos y búsqueda de frecuencias óptimas cuya implementación ha sido proporcionada por el Grupo de Neurocomputación Biológica de la Universidad Autónoma de Madrid (Fernandez-Vargas et al., 2013; Morán García, 2015).

3.2.1 Captura y procesado de datos

En la **captura de datos** de actividad eléctrica cerebral es básica la localización de los electrodos. Los cascos utilizados para realizar electroencefalografías siguen el criterio internacional 10/20 (Trans Cranial Technologies, 2012). Según este estándar, los electrodos se denominan mediante una letra que hace referencia al lóbulo o zona del cráneo en el que se encuentra posicionado el electrodo, tal y como se corresponde en la Tabla 3-1, adaptada de (Trans Cranial Technologies, 2012), y un número de identificación.

Electrodo	Lóbulo
F	Frontal
T	Temporal
C	Central
P	Parietal

Tabla 3-1: Identificación de un electrodo según su lóbulo de localización

Como se puede observar en la Figura 3-1, la localización más favorable para la adquisición de potenciales evocados SSVEP sería la posición Oz, dado que éstos se detectan en la zona occipital del cráneo. Seguida por las posiciones O1, O2 y POz, que se encuentran a una distancia lo suficientemente cercana a Oz, pudiendo ser tomados en cuenta en la captura de datos.

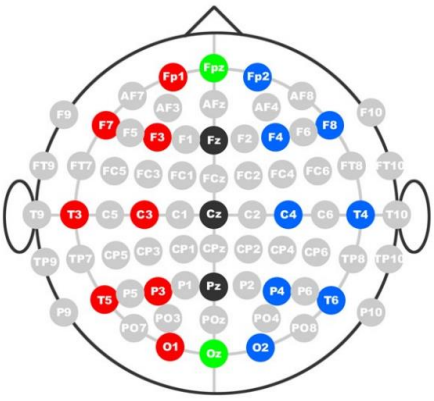


Figura 3-1: Colocación de electrodos según el estándar 10/20

Se ha escogido el casco **Emotiv Epoc** (Emotiv, 2016) como instrumento no invasivo destinado a la obtención de señales mediante EEG, disponible en el laboratorio del Grupo de Neurocomputación Biológica. Se trata de un casco de bajo coste inalámbrico portable, con compatibilidad para sistemas operativos Windows, Linux, OSX, Android e iOS. Proporciona una frecuencia de muestreo de señales de 128 Hz, medida mediante 14 canales y dos electrodos de referencia que siguen el estándar 10-20. Dado que nuestra interfaz-cerebro máquina debe trabajar a tiempo real, la cantidad de muestras que se pueden capturar para su posterior análisis se ve limitada tratando de disminuir los retardos al máximo.

El inconveniente principal de este casco reside en la ausencia de un electrodo en la posición Oz, la más adecuada para la captura de respuestas SSVEP, sin embargo ésta se ve compensada por la existencia de canales en las posiciones O1 y O2, la trivialidad de su colocación y configuración, además de que el propio casco consta de un sistema de ADC (Conversión Analógica/Digital) que acondiciona la señal para ser analizada. No obstante, se tendrán en cuenta estudios previos acerca de la repercusión de la omisión de toma de señales en la posición Oz en la precisión de la señal (Morán García, 2015), para el desarrollo de la aplicación.

En el **proceso de análisis** de los datos recogidos durante la captura de datos mediante EEG, ha de tenerse en consideración que las señales capturadas se ven afectadas por ruido. Con el fin de descartar la mayor cantidad de contaminación de las señales posible, se realiza la grabación de una línea basal o *baseline*, es decir, la monitorización de actividad cerebral que no esté sujeta a ningún tipo de estímulo visual, lo cual nos proporcionará una referencia de la actividad cerebral en reposo de cada individuo. El *baseline* obtenido nos facilita estudiar la relación que existe ente éste y la señal sometida a estímulo visual y, de esta forma, poder obtener un análisis más preciso de los datos de cara a la mejor detección de los SSVEP.

3.2.2 Configuración y selección de frecuencias

Con la intención de exprimir al máximo las contribuciones que proporcionan las interfaces cerebro-máquina y obtener una mayor fiabilidad en los resultados, en este proyecto se busca la máxima adaptación al usuario. Para alcanzarla, se realizará una selección de las frecuencias más óptimas para el usuario a las que se presentan los SSVEP, es decir, aquellas que generan señales más claras y características para cada individuo, mediante la estimulación dependiente de actividad en ciclo cerrado. Se llevará a cabo mediante un proceso de tres fases, comenzando por un barrido por todas las frecuencias de estimulación posibles (Fernandez-Vargas et al., 2013):

Primera fase de barrido de frecuencias: En primer lugar, se establecerá una serie de frecuencias objetivo, incluidas en un rango entre 6 Hz y 20 Hz, frecuencias que se han demostrado utilizables en pruebas anteriores (Morán García, 2015). Durante esta primera fase del ciclo cerrado, se grabará el *baseline* previamente al muestreo de SSVEP con las distintas frecuencias especificadas. Una vez terminado el análisis de las muestras, se estudiarán los resultados: si se encuentran frecuencias que superen un umbral determinado en base a las pruebas, serán consideradas frecuencias aprovechables, y podrán ordenarse de mayor a menor según la puntuación que les ha sido asignada según su relación señal/ruido.

Segunda fase de búsqueda de los mejores pares: En la segunda fase, se busca encontrar las frecuencias óptimas compatibles dos a dos, aquellas con respuesta SSVEP más sólida, para lo cual se compararán las tasas de acierto y error obtenidas al presentar

por pantalla dos frecuencias distintas y pedir al usuario que siga una secuencia predeterminada a la hora de fijar su atención en cada una de ellas.

Tercera fase de mejor compatibilidad entre pares: En la tercera fase, se tomarán aquellos conjuntos de frecuencias con los que se han obtenido los mejores resultados y se compararán entre sí, presentando cuatro estímulos por pantalla simultáneamente, para obtener la mayor compatibilidad posible entre ellos. Finalmente, se almacenarán las cuatro frecuencias óptimas para el usuario que serán utilizadas en la aplicación BCI final.

Existen otros métodos de búsqueda de frecuencias óptimas para el usuario, como el que planteó la Facultad de Tecnología y Biónica de Alemania (Gembler et al., 2015), consistente, en primer lugar, en un estudio de las posibles interferencias entre las frecuencias de la banda alfa (8 Hz - 13 Hz) (Zhu et al., 2010) y las frecuencias objetivo del usuario que se superponen con dicha banda. A continuación, se realiza una estimulación multi-objetivo donde se analizará qué frecuencias obtienen una respuesta SSVEP mayor; y, por último, se volverá a realizar una estimulación con aquellas frecuencias con mejores resultados en la fase anterior. La diferencia entre este método y el empleado en este proyecto reside fundamentalmente en que el primero no hace uso de un algoritmo de búsqueda de ciclo cerrado, con el que se ha confirmado la obtención de resultados que superan los paradigmas de control de SSVEP clásicos (Fernandez-Vargas et al., 2013).

3.2.3 Aplicación de SSVEP-BCI configurable

La aplicación de BCI basada en SSVEP configurable constituye la esencia de este proyecto y por ello se han considerado múltiples necesidades que puedan surgirle al usuario así como distintas perspectivas para afrontarlas y darles solución. Se van a diseñar y desarrollar tres funcionalidades distintas que tratan de proporcionar facilidades a dos de los problemas más significativos y frecuentes que se les presentan a personas que padecen severas discapacidades motoras: dificultades locomotoras y comunicativas.

Con el fin de afrontar las trabas en el **desplazamiento**, se pretende crear una aplicación con la que se podrá dirigir una silla de ruedas o cualquier otro dispositivo que facilite el movimiento, con la ayuda de una pantalla donde el usuario podrá escoger hacia cuál de los cuatro puntos cardinales desea desplazarse.

Por otro lado, se desarrollarán dos aplicaciones distintas para facilitar la **comunicación** con distinto grado de complejidad. Una de ellas, permitirá una comunicación más minuciosa mediante el **deletreo de palabras**, presentando las letras y símbolos del alfabeto divididos en cuatro conjuntos, reduciendo las opciones por cada elección del usuario. Se requiere de tres pasos para completar la selección de una letra, por lo que, pese a su gran concreción, ha de tenerse en cuenta el inconveniente de su lentitud. La otra alternativa permite una comunicación mucho más fluida, aunque significativamente más elemental mediante **ideogramas** presentados al usuario, que atiende a necesidades básicas como las llamadas de urgencia o la comunicación de estados básicos como hambre, sed, comodidad, etc. Esta funcionalidad ha sido diseñada en dos pasos: un primer paso mediante el cual el individuo pueda escoger cuál de dichas necesidades desea satisfacer; y un segundo paso en el que se mostrarán por pantalla opciones más concretas para dar solución a la necesidad seleccionada. Por ejemplo, en el caso del hambre, se mostrarían distintos tipos de comida y el usuario podrá escoger cuál es su preferencia entre ellos.

La aplicación principal presentará, en primer lugar, una pantalla de inicio que le brindará al usuario la posibilidad de **seleccionar la funcionalidad** de la que desea beneficiarse en dicho momento. Además, con motivo de proporcionar una manera al usuario de **volver al menú principal**, se ofrecerá al usuario dicha posibilidad a través de la

simple acción de mantener los ojos cerrados durante un breve lapso de tiempo. De esta manera, cuando la aplicación detecte la frecuencia alfa asociada a los ojos cerrados (~10 Hz) durante 10 recepciones seguidas procedentes del proceso de análisis de la señal, es decir, durante unos 20-25 segundos, el sistema interpretará la voluntad del usuario de acceder al menú de funcionalidad.

3.3 Interfaz gráfica

Con base en los estudios descritos en la sección 2.3.1, el diseño de la interfaz gráfica de las aplicaciones que se van a desarrollar en este proyecto buscará la **flexibilidad de configuración** y **optimización** de todos los parámetros de los estímulos implicados en la respuesta SSVEP obtenida.

3.3.1 Cantidad de estímulos

El diseño de la aplicación permite añadir tantos estímulos como se desee, sin embargo, debido a la limitación de frecuencias óptimas para cada usuario, en las pruebas se ha decidido establecer cuatro estímulos.

3.3.2 Forma y distancia entre estímulos

Dado que la forma no es un parámetro muy relevante para la magnitud de la respuesta SSVEP obtenida, las decisiones de diseño tomadas son puramente estéticas. Las interfaces para las aplicaciones de comunicación no requieren una colocación específica, por lo que los estímulos se verán representados por cuadrados de color o bien imágenes de la misma forma que contendrán un ideograma o un fragmento del abecedario que se desea mostrar. En el caso de la interfaz para la aplicación basada en el control de la dirección para el desplazamiento, se ha optado por utilizar una forma acorde con el uso de los estímulos, de una flecha; y una disposición de los mismos imitando la distribución los cuatro puntos cardinales o direcciones básicas (adelante, derecha, atrás, izquierda).

En el caso de la distancia entre estímulos, pese a que no sea un factor que influya de forma significativa en los potenciales SSVEP, si ésta es muy pequeña, la percepción diferenciada de cada una de las estimulaciones presentadas por pantalla podría verse afectada por la interferencia entre frecuencias captadas por el ojo. Por este motivo, en ambas aplicaciones, se recomienda mantener una distancia media/grande entre estímulos, proporcional a la resolución de pantalla.

3.3.3 Color de los estímulos

El color ha resultado ser influyente en la respuesta SSVEP, obteniendo los mejores resultados haciendo uso de los colores que aporten mayor contraste con el fondo. Por ello, y dado que el fondo de las aplicaciones desarrolladas es negro, las interfaces dedicadas a de la comunicación por deletreo, para realizar una buena distinción entre los estímulos y su contenido, se ha decidido colorear cada estímulo de un color diferente, escogiendo colores que implican un alto contraste con el fondo negro. Se hará uso de la composición de color RGB (*Red, Green, Blue*) en los colores: cian, verde claro, amarillo y magenta, como puede verse en la Figura 3-2 (a), adaptada de (HTML Color Codes, 2016).

En el caso de la dirección de la locomoción, se ha escogido el blanco para colorear las flechas que representan los estímulos, como en el ejemplo mostrado en la Figura 3-2 (b). Puesto que ambos colores son opuestos, el contraste será máximo y se espera obtener los mejores resultados. Mientras que los ideogramas vendrán representados por imágenes correspondiente a cada actividad.

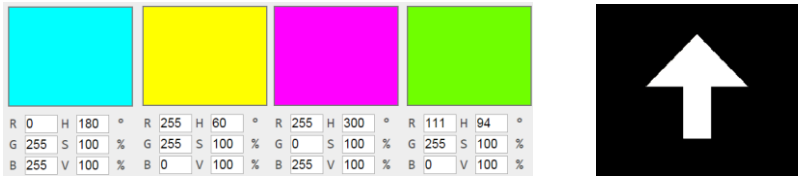


Figura 3-2: Colores RGB escogidos para el deletreo y ejemplo de estímulo indicador de dirección

3.3.4 Tamaño de los estímulos

El tamaño de los estímulos se ha demostrado directamente proporcional a la magnitud de respuesta SSVEP, por lo que se podría inferir que la mejor opción supondría crear los cuatro estímulos principales con el mayor tamaño posible, dentro de los límites de tamaño de la pantalla utilizada. Sin embargo, desde el punto de vista de la usabilidad, pilar básico del proyecto, esto podría producir interferencias entre los distintos estímulos dado que la distancia entre estímulos sería demasiado pequeña y, así mismo, implicaría una mayor probabilidad de extenuación visual para usuario. Por estos motivos, se recomienda un tamaño de estímulo medio en relación a estos factores. Unos 200 píxeles por lado podría resultar un tamaño adecuado a la aplicación, dado que se van a emplear, para la validación de la interfaz, monitores de 23''.

3.3.5 Configuración de la interfaz

Siempre tomando en consideración uno de los objetivos básicos del proyecto: la usabilidad para el usuario, se ha decidido realizar una interfaz fácilmente configurable. Recomendando un tamaño de estímulo medio, como se argumentaba en la sección 3.3.4, el usuario podrá decidir en la elección del tamaño de los estímulos parpadeantes que se mostrarán en la interfaz, la distancia entre ellos, así como el margen izquierdo que debe respetarse entre el linde de la pantalla y el primer estímulo.

3.3.6 Librería

En las interfaces cerebro-máquina con potenciales visuales evocados, como respuesta a la estimulación parpadeante a determinada frecuencia, se genera una resonancia en la zona occipital del cerebro de la frecuencia presentada en el estímulo en el que el usuario fija su atención. Por este motivo, las frecuencias de parpadeo de los estímulos creados por la interfaz gráfica deben ser lo más precisas posibles con respecto a la frecuencia deseada. La librería empleada en el desarrollo de la interfaz gráfica juega, por tanto, un papel transcendental en el rigor de las frecuencias a las que se muestran los estímulos. Con base en un estudio previo a este (Morán García, 2015), se ha escogido la **librería Qt** (The Qt Company, 2016) en su última versión (5.5.1), debido a que, tras varias pruebas con distintas librerías, ésta es la que mejor precisión temporal y capacidad de adaptación al rango de frecuencias que se pretende utilizar, entre 6 Hz y 20 Hz , presenta.

Existen diversos productos y herramientas que facilitan la creación de interfaces gráficas sencillas orientadas a la estimulación visual para las interfaces cerebro-máquina. Entre ellas se encuentran opciones con licencias de pago, como el software desarrollado por g.tec (g.tec Medical Engineering, 2015), aunque tienen el evidente inconveniente no tener libre acceso al código fuente con el que trabajar en investigación. Por otro lado, también hay herramientas de software libre como las implementadas por EEGLAB (Swartz Center for Computational Neuroscience, 2016), OpenBCI (OpenBCI, 2016) y OpenViBE (OpenViBE, 2016). Sin embargo, ninguna de ellas hace uso de la librería Qt, que ha demostrado ser la más eficiente en este tipo de estimulación para las BCI en tiempo real.

4 Desarrollo

4.1 Adquisición de la actividad cerebral

El Emotiv Epoc (Emotiv, 2016) ha sido el casco elegido para la adquisición mediante EEG de la actividad cerebral para este proyecto. Consta de 14 canales de detección de señales y dos referencias, mediante los cuales se registran los datos a 128 Hz.

La comunicación entre el casco y el ordenador se produce de manera inalámbrica y en tiempo real, mediante el uso de su SDK dedicado, cuyas librerías están programadas en el lenguaje de programación C++, en el que está implementado todo el proyecto. Además, permitirá evaluar la calidad de la señal de los electrodos y la grabación de los datos mediante la aplicación *Testbench*. El uso de esta aplicación resulta esencial, ya que la precisión de la interfaz cerebro-máquina que se está desarrollando depende directamente de la calidad de la señal obtenida por el casco, entre otros factores.

La principal ventaja del uso de este casco, como se introducía en la sección 3.2.1, es la facilidad de configuración del mismo en tan solo unos minutos, así como la comodidad para el usuario, pese a la necesidad de impregnar los electrodos con líquido iónico. Sin embargo, y dado que la actividad cerebral detectada por el casco corresponderá a potenciales visuales evocados SSVEP, la carencia de un electrodo colocado en la posición Oz, zona occipital donde se generan este tipo de señales, supone ciertos inconvenientes en la calidad de la respuesta.

Se realizaron distintas pruebas (Morán García, 2015) combinando los distintos canales disponibles, con la intención de subsanar la ausencia del electrodo en la posición Oz, haciendo uso de los electrodos en las posiciones O1 y O2, los más cercanos a éste y colocados en la zona occipital del cráneo, así como de varios electrodos de la parte frontal, con el objetivo de atenuar parte del ruido generado en dicha zona del cerebro. Dichas pruebas contemplaron y trataron, mediante la grabación de una línea basal (*baseline*) previa a la estimulación, el ruido de fondo del cerebro, así como pequeños eventos que afectan a la captura de datos como puede ser el parpadeo involuntario. Se estudió la implicación de cada uno de los hemisferios del cerebro por separado en los registros obtenidos durante la estimulación, sin embargo finalmente se optó por escoger la combinación de ambos hemisferios con mejores resultados, **(O1+O2)-(F3+F4)**, dado que la lateralidad conllevaba demasiada dependencia de la BCI con cada usuario en concreto.

4.2 Procesamiento de las señales

Pese a que no se entrará en detalle de la implementación de las etapas de captura de datos, análisis de la señal obtenida y búsqueda de frecuencias óptimas (Morán García, 2015), que hacen uso de las funciones de la SDK de Emotiv Epoc, es importante adquirir una idea general del funcionamiento de todo el proceso de manera que posteriormente se pueda integrar con la interfaz gráfica diseñada y comprender los resultados obtenidos.

El procesamiento de las señales de este proyecto se corresponde con el expuesto en el esquemático de la Figura 2-1. La fase de eliminación de ruido se corresponde con la fase de *de-trending*, la fase de obtención de características se realiza mediante un enventanado *Hanning* y la posterior clasificación de la señal se lleva a cabo gracias a la aplicación del algoritmo de la Transformada de Fourier Discreta. Estas fases se ven representadas en la Figura 4-1:

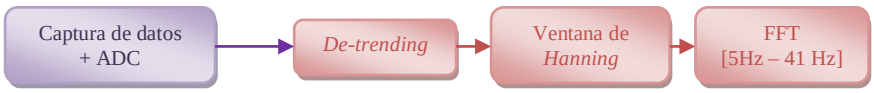


Figura 4-1: Esquema del proceso de análisis de la señal

Dado que el sistema requiere que se capture y analice las señales cerebrales obtenidas en tiempo real, resulta de gran relevancia que éste funcione con la menor latencia posible. Por este motivo, el enventanado de la señal, que permitirá disminuir los efectos de las discontinuidades en la señal, deberá realizarse con ventanas pequeñas, en este caso, con una duración de dos segundos con solapamiento al 50%. Un segundo correspondiente al último de la ventana anterior y el siguiente correspondiente a 128 nuevas muestras, siendo un total de 256 muestras por ventana.

La primera fase del análisis de la señal es el ***de-trending*** que consistirá en la eliminación de tendencias lineales o constantes, de manera que no éstas no se presentarán como ruido de baja frecuencia. Esta fase ayudará a eliminar las nuevas componentes frecuenciales que no se encontraban en la señal original, que se producen al realizar el FFT o algoritmo de la Transformada de Fourier Discreta (DFT) de un enventanado de una señal discreta con componentes lineales.

A continuación, se aplicará la **ventana de *Hanning*** a las muestras recogidas. De este modo, lograremos una buena resolución de la frecuencia buscada, dado que la función de la ventana de Hann concede mayor importancia al lóbulo principal y baja la resolución de las frecuencias que se encuentran en los lóbulos laterales, como puede observarse en el ejemplo de la Figura 4-2 (a). Este tipo de ventana es uno de los más utilizados para el análisis de señal de estas características (Fernandez-Vargas et al., 2013).

Por último, se aplicará la **DFT**, implementada con base en las librerías matemáticas que realizan el algoritmo FFT. Las frecuencias afectadas por la Transformada de Fourier Discreta serán aquellas comprendidas en el rango de 5 Hz a 41 Hz, dado que, pese a que las frecuencias utilizadas en la estimulación se encuentren entre 6 Hz (mínimo necesario para obtener respuestas VEP de estado estacionario) y 20 Hz, resulta muy relevante tener en cuenta también las frecuencias adyacentes y sus armónicos (múltiplos de la frecuencia estimulada). Con ello obtendremos un claro pico de frecuencia, correspondiente a la frecuencia buscada, tal y como puede observarse en la Figura 4-2 (b).

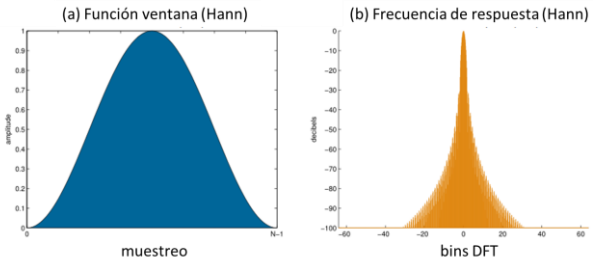


Figura 4-2: Enventanado Hanning de una señal discreta y su función DFT

La comunicación entre los procesos creados mediante la función *fork()*, de la captura de datos del casco y el análisis de la señal se realiza mediante las llamadas al sistema *msgsnd()* y *msgrcv()*, de la librería *<sys/msg.h>*.

4.3 Frecuencias óptimas

Una de las bases del presente proyecto es la búsqueda de la máxima adaptabilidad al usuario, por ello, dentro del rango de frecuencias de estimulación seleccionado entre 6 Hz

y 20 Hz, es importante realizar una optimización de las mismas para cada usuario, esto es, obtener aquellas frecuencias que generan una mejor y más clara respuesta SSVEP a la estimulación, logrando así disminuir al mínimo el error en las funcionalidades de comunicación y desplazamiento implementadas.

Los factores como la amplitud de la señal producida, los armónicos de la frecuencia de estimulación, sus adyacentes y la posible deriva de la misma en estas señales resultan muy dependientes de cada usuario en concreto. Por ello, la misma estimulación ocasionará en cada usuario una reacción diferente e incluso, las respuestas generadas por una misma persona a un mismo estímulo pueden diferir a lo largo del tiempo.

La implementación de búsqueda de las frecuencias óptimas de cada usuario, así como su adaptación al casco Emotiv Epoc, se llevaron a cabo mediante un algoritmo de ciclo cerrado que sigue el esquema desarrollado en la sección 3.2.2 (Fernandez-Vargas et al., 2013; Morán García, 2015). En primer lugar, el algoritmo realiza un **barrido del rango de frecuencias** mencionado, presentando por pantalla una sola estimulación cada 15 segundos; y almacena en un fichero de texto la puntuación de cada una de las frecuencias estimuladas, obtenida mediante la expresión:

$$P_f = X(f)+X(2f)+ 12\ [X(f-1)+X(f+1)+X(2f-1)+X(2f+1)]$$

En dicha expresión se consideran tanto los valores de la transformada de Fourier en la frecuencia fundamental $X(f)$ y en su primer armónico $X(2f)$. El cálculo de las frecuencias adyacentes se realiza reduciendo a la mitad su peso en la puntuación, puesto que de este modo se busca evitar las confusiones entre frecuencias de estimulación muy próximas.

A continuación, el algoritmo realiza una segunda estimulación, aunque esta vez presentando por pantalla **pares de las frecuencias**, obteniendo aquellas frecuencias que presentan mejor compatibilidad entre sí. Tras ello, se inicia una tercera estimulación **combinando** aquellos **pares** con mejores resultados de dos en dos, presentando cuatro estímulos por pantalla. De estas combinaciones, se escogen las cuatro frecuencias más compatibles entre sí para el usuario, que serán los que utilizaremos en los cuatro estímulos necesarios en cada una de las funcionalidades de la aplicación.

4.4 Frecuencias prefijadas

Con motivo de realizar un estudio más exhaustivo referente a las ventajas y mejoras que proporciona la adaptación al usuario, además la optimización de las frecuencias obteniendo, mediante un algoritmo de ciclo cerrado, aquellas que mejor resultados obtienen para el usuario, se realizarán pruebas específicas con frecuencias prefijadas con anterioridad.

Dichas frecuencias serán seleccionadas teniendo en cuenta tres criterios de distinción de frecuencias:

- No hacer uso de frecuencias múltiplos de otras frecuencias, dado que podrían confundirse con los armónicos de las mismas y crear confusión a la hora de distinguir la frecuencia seleccionada.
- La distancia entre las frecuencias ha de ser de al menos 1 Hz, tratando, así, de subsanar el posible error, de décimas de hercio, en la toma de frecuencias.
- Y, por último, es imprescindible que las frecuencias puedan ser representadas por la interfaz gráfica, no excediendo su capacidad de refresco de los estímulos marcado por las características de un monitor de 60Hz.

Las frecuencias escogidas en base a estas pautas, han sido 6 Hz, 8 Hz, 10 Hz y 11 Hz. Ninguna frecuencia es armónico de la otra y todas se distancian entre sí por más de 1 Hz por lo que no existen problemas de confusión.

4.5 Implementación de la interfaz gráfica

4.5.1 Librería Qt

Para realizar la implementación de la interfaz gráfica de este proyecto se han empleado las funciones de la última versión 5.5.1 de **librería Qt** que hace uso del lenguaje de programación C++, en la plataforma de sistema operativo Linux.

Los mecanismos de *signals* y *slots* suponen una de las características principales de Qt, así como la singularidad que mayor diferencia establece entre dicha librería y otras también utilizadas para el desarrollo de interfaces gráficas. Permiten la comunicación entre objetos. En respuesta a una señal que se emite cuando sucede un determinado evento, se llama a un slot de manejo de la señal. Existe tanto la posibilidad de beneficiarse de los slots predeterminados de la librería como añadir slots propios; funcionalidad que resultará conveniente para establecer el parpadeo de los estímulos por pantalla. Todo ello redunda en una mayor precisión temporal de la estimulación.

4.5.2 Estructura de la aplicación

La interfaz gráfica desarrollada en este proyecto sigue la estructura representada en la Figura 4-3, organizada en clases.

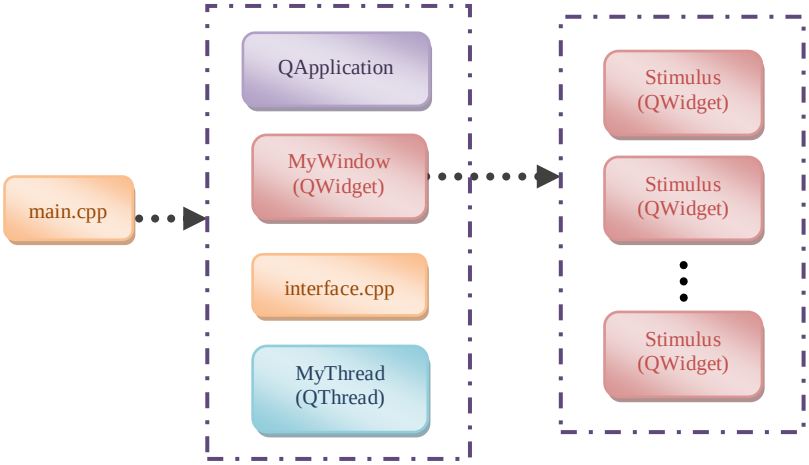


Figura 4-3: Modulado de la interfaz gráfica de la aplicación

La clase **QApplication** es la encargada de guiar el flujo de toda aplicación Qt así como la configuración principal de la misma, por lo que debe ser creada antes que cualquier otro objeto de la interfaz. Engloba el bucle de gestión de eventos, donde se procesan y envían todos los eventos procedentes del sistema de ventanas y otras fuentes. En el caso de este proyecto, únicamente se utilizará su función *exec()*, que ejecutará el bucle de eventos y con ello comenzará la ejecución de la interfaz.

Antes de comenzar con la ejecución de la interfaz, deben definirse y configurarse los objetos o *widgets* que formarán parte de la interfaz de usuario. Para ello se hará uso de las funciones definidas en **interface.cpp**, donde se leerán y tratarán los ficheros de configuración de la aplicación que albergan la información de los estímulos. Dicha configuración se empleará para la inicialización y composición de las clases de **MyWindow** y **Stimulus**, que heredan de la clase **QWidget**, que se encargarán de gestionar

los objetos mostrados por pantalla. Así mismo se hará uso de la clase **QThread** con motivo de crear un hilo de control para el intercambio de mensajes entre distintos procesos.

Las declaraciones de dichas clases encargadas del funcionamiento de la interfaz gráfica pueden encontrarse en el Manual del programador.

4.5.3 Hilo de espera de mensajes

Dada la necesidad de la interfaz de recibir la información obtenida tras el análisis de la señal cerebral obtenida mediante la técnica de la electroencefalografía, para proporcionarle el *feedback* de la respuesta correspondiente al usuario, será necesario crear un hilo de ejecución que trabaje de forma asíncrona al proceso principal de creación de la interfaz. Para lograrlo, se creará una clase específica, **MyThread**, que heredará de la clase **QThread** de Qt y se sobrescribirá su función *run()*, para que realice la tarea deseada. Esta clase está declarada en los ficheros mythread.cpp y mythread.h tal y como se encuentra en la Figura MP-1 del anexo correspondiente al Manual del programador.

Este subproceso será el encargado realizar la recepción de los datos recogidos por el casco. Una vez se reciba un dato, se comparará dicha información de la resonancia SSVEP con la frecuencia de parpadeo de cada uno de los estímulos representados por pantalla, y mediante el envío de un evento del sistema, se le indicará al proceso principal qué estímulo ha sido seleccionado por el usuario. Para ello, se ha asignado a cada estímulo un evento único. En este caso, dado que se disponen de cuatro estímulos seleccionables, se han empleado las cuatro teclas de dirección como identificación del estímulo. El hilo continuará a la espera de un nuevo dato mediante un bucle infinito.

El envío y recepción de mensajes entre los procesos de análisis de la señal y de la interfaz gráfica, se ha implementado mediante las llamadas al sistema *msgsnd()* y *msgrcv()*, de la librería *<sys/msg.h>*, cuya función es la comunicación entre procesos a través de la cola de mensajes del sistema. La implementación de esta comunicación puede encontrarse en el Anexo I.

4.5.4 Ventana

Se ha desarrollado una clase específica, **Mywindow** para gestionar la ventana principal de la interfaz gráfica. Esta clase depende de la clase **QWidget**, que es la base para crear cualquier objeto de la interfaz gráfica. Dispone de funciones que permiten manejar eventos entre distintos procesos incluidas llamadas del sistema, como pulsaciones de teclado o clics con el ratón; así como mostrar todo tipo de imágenes y diseños por pantalla.

La declaración de Mywindow puede encontrarse en el fichero mywindow.h y en la Figura MP-2 del anexo correspondiente al Manual del programador; donde pueden distinguirse las variables y funciones utilizadas para la configuración de los estímulos, definidos en la clase Stimulus.

Se hará uso de la clase QVBoxLayout, que hará las funciones de contenedor de los estímulos, permitiendo posicionarlos en la pantalla. Así mismo, se creará un objeto de la clase QTimer que, al conectarse a una señal de *timeout()*, ejecutará, cuando se cumpla el tiempo establecido, el *slot closeWindow()* que se encargará de cerrar la aplicación de forma automática. Esto facilitará la búsqueda de frecuencias óptimas para el usuario en ciclo cerrado.

Es importante destacar dos de las funciones esenciales en el flujo de la aplicación: *myEvent()* gestionará el envío de las eventos al sistema que permiten la comunicación entre el hilo de espera de mensajes y la aplicación principal, donde se deberá actualizar la

pantalla y los estímulos mostrados con base en la elección realizada por el usuario. Esta tarea se realiza en la función *refreshStimulus()*, donde se actualizarán los parámetros de los estímulos teniendo en cuenta el tipo de funcionalidad que se esté ejecutando. El código de implementación de dicha función puede encontrase en el Anexo II.

4.5.5 Estímulos

La clase **Stimulus** hereda, así mismo, de la clase **QWidget**, permitiendo crear tantos objetos Stimulus como estímulos se deseen presentar en la ventana principal, en la cual se incluyen e inicializan. La declaración de Stimulus puede encontrarse en el fichero stimulus.h y en la Figura MP-3 del anexo correspondiente al Manual del programador.

Se utilizará la clase QLabel con la intención de que, mediante una serie de variables, se pueda establecer la posición y tamaño de los estímulos, así como su contenido, ya sea como un cuadro de color o una imagen.

A la hora de implementar las funcionalidades del BCI, se ha decidido realizar la implementación de dos aplicaciones alternativas para la comunicación mediante deletreo, con el objetivo de estudiar la eficiencia de ambas y las ventajas y desventajas que cada una de ellas presenta. Una de ellas se realizará mediante la presentación de cuadros de color parpadeantes por pantalla que contendrán en su interior una cadena de caracteres del alfabeto, que se verá modificada en la función *refreshStimulus()* de la clase Mywindow, realizando las reparticiones del abecedario correspondientes a la selección del usuario, reduciendo sus posibilidades en cada elección hasta obtener la letra deseada. Esta versión aporta mayor universalidad a la aplicación ya que bastaría con definir el alfabeto deseado mediante cadenas de caracteres para ser compatible en cualquier idioma. La otra alternativa de desarrollo de dicha funcionalidad es similar al método utilizado para las funcionalidades de desplazamiento y comunicación mediante ideogramas: la presentación de imágenes por pantalla. En este caso no se hará uso de cuadros de color y cadenas de texto, sino que la propia imagen ya tendrá un color de fondo y el texto deseado incrustado en ella, por tanto se necesitarán tantas imágenes como posibles opciones se presenten por pantalla.

Para la gestión de las imágenes se utilizará la clase QPixmap, dado la carga de imágenes resulta más eficiente y rápida que la que ofrecen otras clases similares; lo que supone más precisión en el parpadeo y mayor exactitud en la calidad de la respuesta SSVEP obtenida. Mientras que en el caso de los cuadros de color, utilizados en la funcionalidad de la comunicación mediante el deletreo, se crearán con base en la clase QPalette, donde se podrá especificar tanto el color del fondo del estímulo como del texto mostrado en su interior. Además, se guardará el valor de la frecuencia a la que debe producirse el parpadeo de cada estímulo, gestionado por el slot *blink()*, que se ejecutará cuando se envíe una señal una vez que el objeto QTimer conectado a ella cumpla el tiempo establecido: medio periodo, magnitud inversa a la frecuencia.

Se pueden distinguir tres tipos de estímulos que se identificarán mediante las variables lógicas *ind* y *fix*.

Los **estímulos principales** son representados por pantalla como imágenes o cuadros de color parpadeantes a una frecuencia determinada, ofreciendo las distintas posibles elecciones disponibles en la aplicación al usuario.

Los **estímulos indicadores** (*ind*) serán los encargados de mostrar al usuario por pantalla el *feedback* de la respuesta SSVEP obtenida. En el caso de las aplicaciones que tratan con imágenes, como la aplicación orientada al control del desplazamiento, se dibujará por pantalla un recuadro rojo enmarcando el estímulo seleccionado; mientras que en el caso de

la aplicación de deletreo, se presentará en el centro de la pantalla la palabra que se esté formando y actualizando cada vez que se reciba el evento de pulsación virtual que se envía al recibir la respuesta SSVEP procedente del proceso de gestión del casco EEG.

Los **estímulos del barrido de frecuencias** (*fix*) son los más sencillos de la aplicación. Únicamente mostrarán una pequeña imagen bajo uno de los estímulos principales al inicio de la ejecución. De este modo, el usuario podrá tomarlo como punto de referencia y sabrá en qué estímulo deberá focalizar su atención durante el entrenamiento.

4.5.6 Ficheros de configuración de la interfaz

Los parámetros bajo los cuales deben ser construidos los estímulos vendrán dados mediante ficheros de configuración (.cfg). Serán necesarios tantos ficheros de texto de configuración como funcionalidades distintas tenga el programa. Esta característica le proporciona gran flexibilidad a la aplicación, dada la facilidad de incorporación de nuevas funcionalidades a la misma en caso de ser necesarias.

Todos los ficheros constan de un esquema común en las dos primeras líneas del archivo, donde se especifica el identificador del tipo de funcionalidad, cuyas posibilidades están representadas en la Tabla 4-1, que se va a configurar con el mismo y el número de estímulos principales que se presentará por pantalla al usuario.

Identificador de funcionalidad	Descripción de configuración
0	Menú de inicio de selección de funcionalidad
1	Funcionalidad de desplazamiento mediante flechas
2	Funcionalidad de comunicación mediante deletreo con cuadros de color
3	Funcionalidad de comunicación mediante deletreo con imágenes
4	Funcionalidad de comunicación mediante ideogramas

Tabla 4-1: Identificación de los tipos de funcionalidades de la aplicación

Como se puede apreciar en la Figura 4-4, la estructura de los ficheros de configuración del menú de selección de funcionalidad y del desplazamiento es la misma. Tras las dos primeras líneas, comunes a todos los ficheros de configuración, en cada una de las líneas siguientes se encuentra la información necesaria para la creación de cada uno de los estímulos.

En estos casos, el estímulo vendrá representado por una imagen, ya sea una flecha en el caso del desplazamiento o un cuadro de color con el nombre de la funcionalidad en caso del menú. A continuación de la ruta de la imagen, encontramos varios parámetros numéricos que representan, en orden de aparición, la frecuencia de parpadeo del estímulo, su posición con respecto al margen izquierdo, su posición respecto al borde superior, y su tamaño de ancho y alto.

```
Type: 0
Num Stimulus: 4
Images/desplazamiento1.png, 6, 825, 200, 300, 175
Images/deletreo1.png, 9, 825, 700, 300, 175
Images/fondo_transparente.png, 15, 1000, 1000, 175, 175
Images/fondo_transparente.png, 18, 1000, 1000, 175, 175
```

```
Type: 1
Num Stimulus: 4
Images/flechaarriba.png, 6, 825, 100, 200, 200
Images/flechaderecha.png, 9, 1200, 375, 200, 200
Images/flechabajo.png, 15, 825, 650, 200, 200
Images/flechaizquierda.png, 18, 450, 375, 200, 200
```

Figura 4-4: Ejemplo de configuración de la funcionalidad del menú de inicio y desplazamiento

En el caso de la configuración de la funcionalidad de comunicación mediante deletreo, debido a que su diseño permite seleccionar una letra en tres pasos, cada uno de ellos con sus respectivas posibles pantallas, se ha decidido implementar dos tipos de estimulación visual para alcanzar dicho objetivo. Una de las versiones creará dichos estímulos mediante cuadros de color, mientras que la otra utilizará imágenes referenciadas en el fichero de configuración.

En la Figura 4-5 se muestra el fichero de configuración de deletreo mediante cuadros de color, donde en cada línea referente a un estímulo diferente incluirá la división del abecedario de tamaño análogo a las demás, que se presentará por pantalla durante el primer paso de la selección de cada letra, así como el color deseado para el estímulo, indicado en los tres últimos parámetros de cada línea mediante el formato RGB.

```
Type: 2
Num Stimulus: 4
abcdefghijklmnop¬ 13, 300, 100, 175, 175, 255, 0, 255
qrstuvwxyz012345¬ 10, 875, 100, 175, 175, 255, 255, 0
6789.,(<;:*)>'?^¬ 19, 300, 475, 175, 175, 0, 255, 255
]\"[{}+-*&/=@%#$~¬ 8, 875, 475, 175, 175,111, 255, 0
```

Figura 4-5: Ejemplo de configuración de la funcionalidad de deletreo con cuadros de color

En la Figura 4-6 puede verse un fragmento del fichero de configuración de la funcionalidad de deletreo mediante imágenes, correspondiente a la información necesaria para la creación y refresco de uno de los estímulos. Como en el caso anterior, se muestra el abecedario correspondiente a cada estímulo durante el primer paso de la selección de una letra, ya que será necesario para indicarle al usuario por pantalla la palabra que se está formando, esta vez acompañado por la ruta de imagen correspondiente a dicho grupo de símbolos y letras. Las líneas siguientes representan las distintas imágenes que deberán emplearse para la actualización de los estímulos después de cada selección por parte del usuario.

```
Type: 3
Num Stimulus: 4
abcdefghijklmnop¬ Images/abc/abc-a.png, 13, 300, 100, 175, 175
Images/abc/abc1-a.png
Images/abc/abc2-a.png
Images/abc/abc3-a.png
Images/abc/abc4-a.png
Images/abc/abc1-1-a.png
Images/abc/abc1-2-a.png
Images/abc/abc1-3-a.png
Images/abc/abc1-4-a.png
Images/abc/abc2-1-a.png
Images/abc/abc2-2-a.png
Images/abc/abc2-3-a.png
Images/abc/abc2-4-a.png
Images/abc/abc3-1-a.png
Images/abc/abc3-2-a.png
Images/abc/abc3-3-a.png
Images/abc/abc3-4-a.png
Images/abc/abc4-1-a.png
Images/abc/abc4-2-a.png
Images/abc/abc4-3-a.png
Images/abc/abc4-4-a.png
```

Figura 4-6: Ejemplo de configuración de la funcionalidad de deletreo con imágenes

Por el contrario, para llevar a cabo una comunicación mediante ideogramas, serán necesarias tantas imágenes como opciones vayan a ser presentadas por pantalla como estímulos. Esta funcionalidad consta de dos elecciones. Una primera elección de la necesidad o estado básico que el usuario desea comunicar y otra donde podrá escoger, entre las cuatro posibilidades presentadas, cómo prefiere solventarla. En la Figura 4-7, pueden distinguirse, en cada línea correspondiente a la parametrización de cada estímulo, cinco rutas de imágenes. La primera correspondería a la imagen del estímulo presentado en el primer paso de la funcionalidad, y las cuatro siguientes atañen a las opciones que se mostrarían por pantalla en caso de que el usuario seleccionara dicho estímulo durante la primera elección. Un ejemplo de este tipo de comunicación puede verse en la Figura MU-5 del Manual de usuario.

```
Type 4
Num Stimulus: 4
Images/hambre.png, Images/desayuno.png, Images/comida.png, Images/cena.png, Images/merienda.png, 6, 350, 100, 250, 250
Images/sed.png, Images/agua.png, Images/refresco.png, Images/zumo.png, Images/vino.png, 9, 1300, 100, 250, 250
Images/llamada.png, Images/112.png, Images/911.png, Images/091.png, Images/061.png, 15, 350, 750, 250, 250
Images/comodidad.png, Images/sillon.png, Images/cama.png, Images/sofa.png, Images/taburete.png, 18, 1300, 750, 250, 250
```

Figura 4-7: Ejemplo de configuración de la funcionalidad de comunicación mediante ideogramas

5 Integración, pruebas y resultados

5.1 Validación de la interfaz gráfica

Previa a la integración con las etapas de captura, análisis y optimización de frecuencias cuya implementación ha sido aportada por el Grupo de Neurocomputación Biológica, se van a realizar una serie de pruebas de validación de la interfaz gráfica desarrollada.

En primer lugar, se realizará una prueba haciendo uso de un fotodiodo con el fin de asegurar que la frecuencia real del parpadeo de los estímulos presentados por pantalla se corresponde con la frecuencia especificada por el fichero de configuración. Así mismo, se va a comprobar el buen funcionamiento de la comunicación entre el proceso principal de la interfaz gráfica y el hilo receptor de mensajes procedentes del proceso de análisis de la señal.

5.1.1 Fotodiodo

Para la realización de la prueba de **validación de las frecuencias de parpadeo de los estímulos** se hará uso de un **fotodiodo** que registra con precisión la estructura temporal del parpadeo lumínico que produce la alternancia entre oscuridad e iluminación. Esto permitirá grabar gran cantidad de muestras de la señal por segundo y almacenarlas en un fichero de datos, que posteriormente analizaremos.

Se ejecutará la interfaz gráfica presentando por pantalla los cuatro estímulos principales de la misma con frecuencias de parpadeo comprendidas en el rango de 6 Hz a 20 Hz. Las frecuencias escogidas para este experimento han sido 6 Hz, 10 Hz, 13 Hz y 20 Hz. Y se posicionará el fotodiodo frente a cada uno de los estímulos, muy próximo a la pantalla, de modo que la iluminación ambiente no influya en la señal medida. El muestreo obtenido de cada estimulación y registrado a través de una tarjeta de adquisición de datos se grabará en un fichero distinto. La representación gráfica de señal obtenida y de su Transformada Rápida de Fourier (FFT) revelarán la frecuencia real de parpadeo de los estímulos.

El experimento se ha llevado a cabo tanto para la estimulación con imágenes, como para el parpadeo de cuadros de color. Los resultados obtenidos para las distintas estimulaciones realizadas con los cuadros de color se presentan en la Figura 5-1, representando las frecuencias 6 Hz, 10 Hz, 13 Hz y 20 Hz, respectivamente:

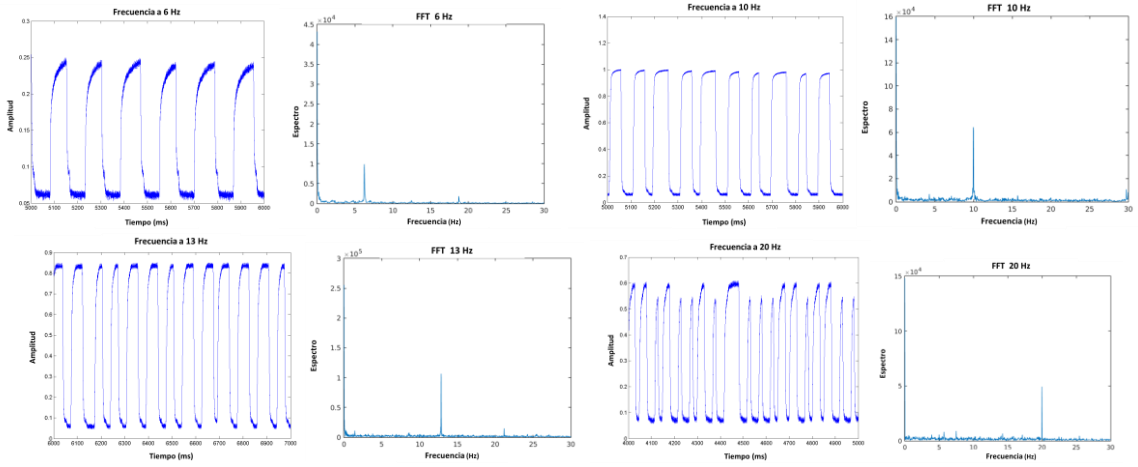


Figura 5-1: Cuadros de color: Representación gráfica de la señal y su FFT

Los resultados obtenidos para las distintas estimulaciones realizadas con imágenes se presentan en la Figura 5-2, representando las frecuencias 6 Hz, 10 Hz, 13 Hz y 20 Hz, respectivamente:

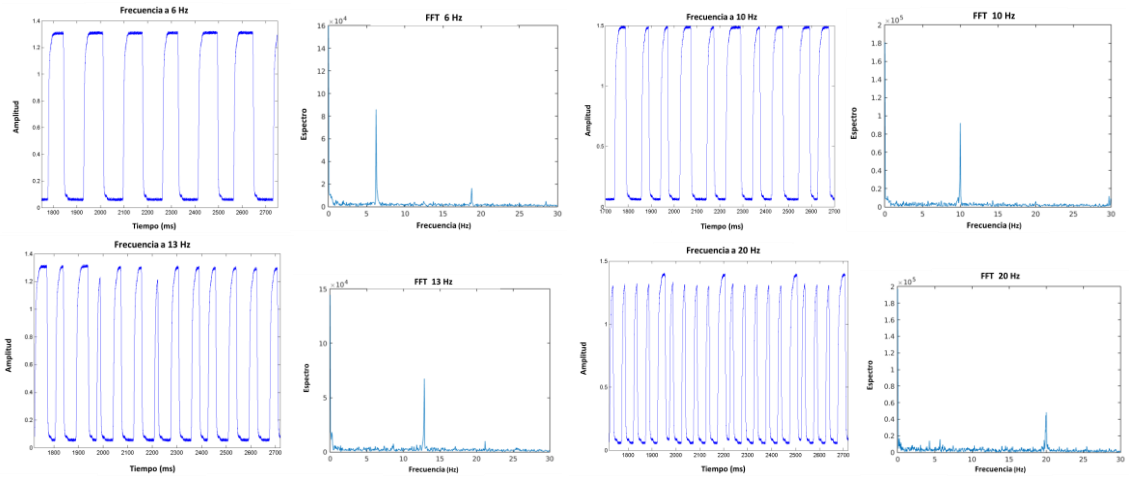


Figura 5-2: Imágenes: Representación gráfica de la señal y su FFT

En ambos casos, los resultados de las pruebas son suficientemente precisos con respecto a las expectativas para las frecuencias 6, 10 y 13 Hz. Sin embargo, a los 20 Hz se empiezan a encontrar ciertas irregularidades.

En el caso de los estímulos representados mediante imágenes, se obtiene un buen comportamiento en el parpadeo a la frecuencia de 20 Hz, debido a que, como se mencionó en la sección 4.5.5, se hace uso de la clase QPixmap para la carga de las figuras y al tratar con bitmap, el cambio de imagen se produce de manera muy eficiente.

Sin embargo, en el caso los estímulos creados mediante cuadros de color haciendo uso de la clase QPalette, la eficiencia de repintado de la pantalla es menor y produce fallos a frecuencias cercanas a los 20 Hz. Esto significa que el rango de posibles frecuencias de estimulación encuentra su límite en una frecuencia más baja, además de generar una señal mucho más inestable.

Con esta prueba podemos empezar a distinguir las principales diferencias y características de las dos alternativas implementadas de la comunicación mediante deletreo. En el caso de los cuadros de color, se debe sacrificar el uso de ciertas frecuencias y cierta precisión en la señal, sin embargo, la implementación mediante, únicamente, la variación del texto presentado en los estímulos permite una gran simplificación de dicha funcionalidad, descrita en la sección 4.5.5, así como alcanzar una gran universalidad y flexibilidad, dado que al no hacer uso de imágenes predefinidas, bastaría la simple definición previa del alfabeto que se va a utilizar para que la aplicación pueda ejecutarse en cualquier idioma. Por otro lado, el deletreo haciendo uso de imágenes, aunque implica cierta complejidad en la gestión del refresco de los estímulos tras cada una de las selecciones, permite al usuario utilizar un rango de frecuencias más amplio, ofreciéndole más posibilidades de adaptación así como una mejor eficiencia en la señal generada, dado que la estimulación resulta más estable.

5.1.2 Validación del control de la interfaz

Una vez hemos comprobado la concordancia de la estimulación con las frecuencias especificadas, es importante verificar que la comunicación entre los subprocesos del programa de la interfaz se establece y se desempeña correctamente.

El hilo de espera de mensajes, una vez recibe el dato de la frecuencia analizada de la actividad cerebral que ha sido capturada y procesada, debe informar al proceso principal de la interfaz que controla los widgets presentados por pantalla qué estímulo ha sido seleccionado por el usuario. Para ello, se hace de la funcionalidad de gestión de eventos de teclado de Qt.

La **validación de la comunicación entre subprocesos** se ha implementado omitiendo en el hilo de ejecución la funcionalidad de recepción de mensajes del proceso de control del casco EEG, y enviando virtualmente un evento específico y predefinido manualmente al proceso principal de la interfaz. Si el estímulo principal seleccionado mediante el estímulo indicador coincide con el que está unívocamente relacionado con el evento enviado, la prueba ha finalizado con éxito.

En la Figura 5-3, se representa un ejemplo del código de la prueba realizada para la validación de la comunicación entre subprocesos. Se está enviando evento para la selección del estímulo con id 1, segundo estímulo principal según su orden de creación, siendo éste el widget situado abajo a la izquierda.

```
void MyThread::run(){

    Mywindow * w = Mywindow::getInstance();

    /** Se selecciona el tipo de funcionalidad - DELETREO */
    sleep(6);
    //Envio del evento del segundo estímulo (con identficador 1)
    w->myEvent(1)

    /** Se selecciona el segundo estímulo con ID 1 */
    sleep(6);
    //Envio del evento del segundo estímulo (con identficador 1)
    w->myEvent(1)
```

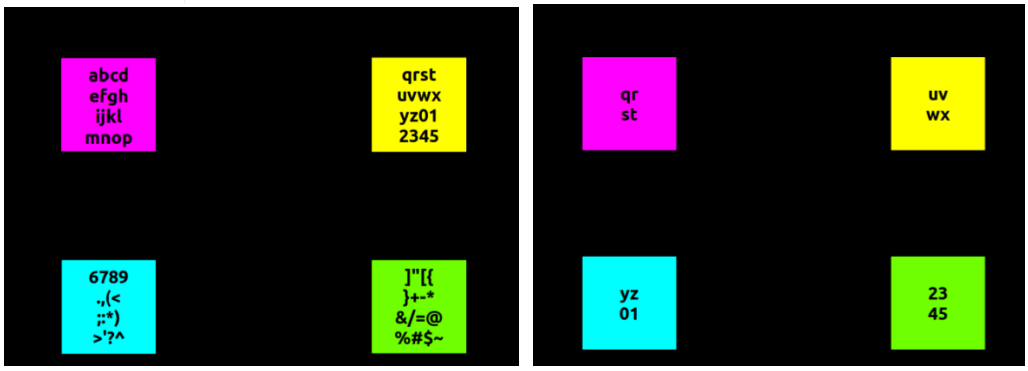


Figura 5-3: Ejemplo de validación de la comunicación entre subprocesos

5.2 Integración con la captura y análisis de señal

La integración de la interfaz gráfica desarrollada con el resto de etapas en las que se estructura una interfaz cerebro-máquina supone uno de los procesos claves del proyecto. Para ello, debe establecerse la comunicación entre los procesos de análisis de la señal y la interfaz gráfica. Como se menciona en la sección 4.5.3, esta comunicación se realiza mediante las llamadas al sistema *msgsnd()* y *msgrcv()*, haciendo uso de una cola de mensajes.

Antes de comenzar el proceso de captura de la señal, se crean dos identificadores de cola de mensajes mediante la función *msgget()*. Uno de estos identificadores determinará la comunicación entre los procesos de captura y análisis de la señal y el otro, servirá de puente para el envío y recepción de las frecuencias procesadas de los potenciales SSVEP,

obtenidos de la resonancia producida en la zona occipital del cerebro del usuario, entre los procesos de análisis e interfaz gráfica, respectivamente.

Ambos procesos deben conocer dicho identificador de cola, por lo que una vez creado, se ejecutará el proceso de la interfaz mediante una llamada a *execl()*, pasándole como parámetros los ficheros de configuración de los estímulos, tal y como se expone en la sección 4.5.6, así como el identificador de cola y las cuatro frecuencias a las que se realizarán las estimulaciones por pantalla:

```
// Transformarmos los argumentos a pasar a la interfaz gráfica a char
char argumento0[DIM_ARGUMENTOS];
char argumento1[DIM_ARGUMENTOS];
char argumento2[DIM_ARGUMENTOS];
char argumento3[DIM_ARGUMENTOS];
char argumento4[DIM_ARGUMENTOS];
sprintf(argumento0, "%d", idColaInterfaz);
sprintf(argumento1, "%d", procesoDatos.frecuenciasAnalisis[0]);
sprintf(argumento2, "%d", procesoDatos.frecuenciasAnalisis[1]);
sprintf(argumento3, "%d", procesoDatos.frecuenciasAnalisis[2]);
sprintf(argumento4, "%d", procesoDatos.frecuenciasAnalisis[3]);

if (execl(ARCHIVO_PROGRAMA_INTERFAZ,ARCHIVO_PROGRAMA_INTERFAZ,ficheroEstimulos_inicio, ficheroEstimulos_mov, ficheroEstimulos_spell,
          argumento0, argumento1, argumento2, argumento3, argumento4, NULL)== -1)
    cout << "    \e[1;35m[WARNING]\e[0m \e[0;37m Fallo en la llamada al proceso de control del robot\e[0m" << endl;
```

Figura 5-4: Ejecución de la interfaz gráfica pasando como parámetro el identificador de cola de mensajes

Una vez ambos procesos han establecido la comunicación y se ha analizado la señal capturada por el casco EEG, se enviará la información de la frecuencia obtenida a la interfaz:

```
// Enviamos el dato obtenido a la interfaz
mensaje.tipoComunicacion = IPC_INTERFAZ_DATOS;
mensaje.ComandoRobot = datoFrecuencia;
errorTransmision = msgsnd(idColaInterfaz, &mensaje, sizeof(mensaje), MSG_NOERROR);
if (errorTransmision < 0)
    cout << "    \e[1;35m[WARNING]\e[0m \e[0;37m Fallo al enviar información a la cola de la interfaz\e[0m" << endl;
```

Figura 5-5: Envío de la frecuencia obtenida de los potenciales SSVEP a la interfaz gráfica

Este dato será recibido por el hilo creado en el proceso de la interfaz gráfica, dónde se realizan las comprobaciones pertinentes, descritas en la sección 4.5.3, para determinar qué estímulo fue seleccionado por el usuario.

5.3 Pruebas y resultados de la BCI

5.3.1 Búsqueda de frecuencias óptimas para cada usuario para la interfaz gráfica implementada

Una vez realizada la integración de la interfaz gráfica con las etapas de captura y análisis de señal, puede dar comienzo la fase de pruebas de integración que se llevara a cabo con una muestra de 8 sujetos voluntarios.

En primer lugar, se realizará la **grabación de un *baseline***, es decir, la grabación de la actividad de fondo del cerebro observando el monitor de la pantalla en negro, sin estimulación, durante 60 segundos. Movimientos musculares, sobre todo los faciales, o parpadeos pueden introducir gran cantidad de ruido en la señal grabada, por lo que se les pedirá a los voluntarios que los eviten en la medida de lo posible.

Una vez obtenido el *baseline*, se realizará un **entrenamiento** con motivo de encontrar las **frecuencias óptimas** para cada usuario. Durante la primera fase, se realizará un barrido de frecuencias, entre 6 y 20 Hz. Se presentará un único estímulo por pantalla al que deberá mirar durante 10 segundos y dispondrá de 2 segundos de descanso para parpadear antes de

estimular la siguiente frecuencia. A continuación, tras 10 segundos de descanso, se presentarán por pantalla pares de estímulos y el individuo deberá centrar su atención en cada uno de ellos sucesivamente, apuntado por un pequeño indicador debajo del estímulo. Por último, después del breve lapso de tiempo de descanso para el usuario, se repetirá el proceso con cuatro estímulos por pantalla. De esta manera, obtendremos los cuatro estímulos más compatibles entre sí para cada usuario.

Entre todas las utilizadas durante el barrido, sólo podemos considerar frecuencias válidas a aquellas cuyas puntuaciones superan un umbral, en este caso establecido en 7, elegido con base en pruebas anteriores (Morán García, 2015). En la Tabla 5-1 se muestran ordenadas de mayor a menor según su puntuación obtenida:

Usuario	# frecuencias válidas	Frecuencias válidas	Frecuencias finales elegidas
1	8	6, 10, 7, 8, 9, 11, 13, 19	13, 8, 9, 6
2	7	9, 20, 8, 19, 11, 7, 13	19, 9, 7, 11
3	4	20, 16, 19, 17	20, 16, 19, 17
4	9	6, 20, 15, 19, 7, 17, 18, 12, 13	15, 6, 13, 18
5	4	7, 10, 16, 6	7, 10, 16, 6
6	4	10, 13, 11, 14	10, 13, 11, 14
7	8	6, 7, 13, 10, 15, 9, 11, 16	6, 15, 10, 7
8	10	6, 15, 11, 8, 13, 17, 14, 20, 19, 18	6, 8, 15, 17

Tabla 5-1: Frecuencias elegidas para cada usuario

Una vez se han obtenido las frecuencias más adecuadas para cada usuario, se realizarán pruebas específicas con la interfaz implementada y sus distintas funcionalidades.

Partiendo de la base de que el algoritmo de captura y análisis de la señal constaba de una precisión máxima en torno al 60-70 % de tasa de acierto en los primeros experimentos tras su desarrollo (Morán García, 2015), puesto que se está trabajando con un casco EEG de bajo coste, en una primera instancia, se decidió realizar las pruebas haciendo uso de una doble comprobación de las frecuencias recibidas con las frecuencias de los estímulos presentados por pantalla, buscando una mayor fiabilidad en los resultados.

Las pruebas diseñadas en primera instancia, se basaban en el seguimiento de varias secuencias, alternando las distintas opciones presentadas al usuario, en el caso de la funcionalidad para el desplazamiento y la comunicación mediante ideogramas, mientras que para el caso de la comunicación mediante deletreo se planteó el deletreo de palabras sencillas de tres letras, tratando de hacer más amenas las pruebas. Se decidió analizar el porcentaje de acierto con respecto al total de selecciones realizadas por cada usuario, siendo en todos los casos un total de entre 15 y 20 selecciones.

Dichos experimentos resultaron en gran medida positivos para las pruebas realizadas sobre las funcionalidades implementadas mediante el parpadeo de imágenes (funcionalidades de desplazamiento y comunicación mediante ideogramas). Sin embargo, en la funcionalidad de deletreo, la tasa de acierto obtenida por cada letra acertada es muy baja. Dicho comportamiento de la interfaz cerebro-máquina tiene su explicación primordial en la limitación de la precisión del algoritmo de captura de la señal para el casco utilizado para la realización de las pruebas: Emotiv Epoc. Como comentábamos anteriormente, durante los experimentos desarrollados con motivo de probar las etapas de captura y análisis de la señal, la tasa de acierto obtenida por cada selección se encuentra alrededor de un 60-70% y, dado que la funcionalidad de deletreo desarrollada requiere de tres

selecciones hasta llegar a la elección de una letra, el error en cada selección se va acumulando, dando así como resultado una tasa de acierto de letra muy baja. Por este motivo, se decide estudiar la tasa de acierto obtenida en cada selección. Los resultados de dicha experimentación reflejan en la Tabla 5-2.

Usuario	% Acierto Imágenes	% Acierto Deletreo: Cuadros de Color	% Acierto Deletreo: Imágenes	% Acierto medio
1	62.07 %	56.6 %	63.52 %	60.73 %
2	57.5 %	48.71 %	60.54 %	55.58 %
3	43.48 %	27.27 %	41.28 %	37.34 %
4	42.66 %	38.53 %	47.75 %	42.98 %
5	60 %	51. 22 %	65.94 %	59.05 %
6	52.45 %	44.11 %	54.61 %	50.39 %
7	53.33 %	45.45 %	52.96 %	50.58 %
8	66.67 %	47.36 %	68.47 %	60.83 %

Tabla 5-2: Resultados finales de estimulación con imágenes y cuadros de color

A la vista de los resultados obtenidos, podemos concluir que los resultados son mejores en el caso de la estimulación mediante imágenes que mediante cuadros de color. Mientras que los resultados de las funcionalidades de desplazamiento e ideogramas superan en la mayoría de los casos una tasa de acierto del 50%, logrando incluso obtener resultados muy cercanos a un 70%, la mayor parte de las pruebas realizadas con el parpadeo de cuadros de color no alcanzan el 50% de tasa de acierto. La precisión en los resultados de los experimentos con las pruebas de color es notablemente peor que con las imágenes, tal y como podíamos prever tras las primeras pruebas de parpadeo de la interfaz con los fotodiodos, ya que, como se comentaba en la sección 5.1.1, la capacidad de alternancia entre imágenes es mucho más rápida y eficaz que el repintado de la pantalla con color, aunque ésta última aporta la ventaja de la flexibilidad de modificación del idioma sin necesidad de crear nuevas imágenes para lograrlo.

En una primera instancia los resultados pueden parecer insólitos dado que los porcentajes de acierto obtenidos en cada usuario no son demasiado homogéneos. Esto se debe a la variabilidad de los usuarios para adaptarse a las señales estimuladas y la generación de la resonancia correspondiente en su cerebro. Puesto que no todos los individuos son capaces de adaptarse y alcanzar un control sobre una interfaz cerebro-máquina, es importante encontrar al menos un sujeto que pueda controlar de manera satisfactoria el BCI. En este proyecto el sujeto con mejor adaptación a este tipo de interfaces cerebro-máquina resultó ser el usuario 8.

El problema encontrado con respecto a la baja tasa de acierto por cada letra, fruto de tres selecciones consecutivas, presentada por pantalla correctamente, se vería fácilmente solventado haciendo uso de un método más preciso para la toma y el análisis de las señales SSVEP generadas por el usuario. Esta solución ya está tomando forma actualmente, puesto que se ha comenzado a realizar la integración de la interfaz gráfica implementada en este proyecto con los métodos de captura y análisis de la señal de un nuevo casco EEG de mayor precisión (g.tec, g.GAMMAcap), adquirido por el Grupo de Neurocomputación Biológica de la Universidad Autónoma de Madrid.

5.3.2 Prueba del sistema BCI con frecuencias prefijadas

Para finalizar, compararemos los resultados obtenidos mediante las pruebas documentadas en la sección anterior con las frecuencias optimizadas para cada usuario y la

misma experimentación realizada con las frecuencias prefijadas establecidas en la sección 4.4 (6 Hz, 8 Hz, 10 Hz, 11 Hz).

FRECUENCIAS PREFIJADAS				FRECUENCIAS ÓPTIMAS
Usuario	**% Acierto - Imágenes**	**% Acierto - Cuadros**	**% Acierto medio**	**% Acierto medio**
1	52.94 %	46.73 %	49.83 %	60.73 %
2	38.81 %	22.2 %	30.51 %	55.58 %
3	34.82 %	16.67 %	25.74 %	37.34 %
4	37.5 %	32.17 %	34.82 %	42.98 %
5	41.17 %	38.2 %	38.69 %	59.05 %
6	33.34 %	28.83 %	31.09 %	50.39 %
7	44.78 %	40.12 %	42.45 %	50.58 %
8	55.56 %	47.91 %	51.73 %	60.83 %

Tabla 5-3: Comparación entre los resultados con frecuencias prefijadas y con frecuencias óptimas

Como esperábamos y puede observarse en la Tabla 5-3 de resultados, mostrando un único porcentaje medio calculado a partir de los obtenidos en cada una las aplicaciones que hacen uso de imágenes, la tasa de acierto tras las pruebas haciendo uso de las frecuencias prefijadas desciende significativamente en comparación con los datos obtenidos con las frecuencias adaptadas al usuario.

6 Conclusiones y trabajo futuro

6.1 Conclusiones

El objetivo principal del proyecto ha sido el desarrollo de una interfaz gráfica fácilmente configurable y adaptable al usuario para la estimulación visual de una interfaz cerebro-máquina basada en SSVEP registrados mediante EEG. Esta herramienta intenta dar una solución a distintos problemas de comunicación y desplazamiento que se manifiestan en personas con movilidad reducida y discapacidades motoras.

Para llevarlo a cabo, se ha realizado, en primer lugar, **un estudio y selección de las distintas técnicas**, como los métodos de captura de las señales y el origen de las mismas, y parámetros, como el color, forma, distancia entre estímulos, etc., influyentes en la usabilidad y adaptabilidad de la interfaz de los cuales se ha hecho uso para su desarrollo. Se ha abordado el diseño de tres funcionalidades que sirvan en aplicaciones reales, abarcando distintos aspectos de la comunicación y el control de dispositivos. Concretamente se ha diseñado una aplicación que permite la comunicación mediante el deletreo, otra para comunicar en necesidades básicas representadas mediante ideogramas.

En lo referente al **desarrollo** de dichas funcionalidades, se ha optado por el uso de una librería orientada a la implementación de interfaces (Qt) mediante la que, gracias a sus características específicas para el tratamiento de señales temporales, el control del flujo de la aplicación, así como del parpadeo de los estímulos presentados por pantalla puede manejarse de manera muy eficiente. En esta fase, la mayor complejidad se ha encontrado en la actualización de los estímulos presentados tras las diversas selecciones realizadas por el usuario, ya que en las tres funcionalidades así como en el menú principal, el cambio de pantalla debe realizarse mediante distintos métodos; siendo el del deletreo el más complejo debido a la repartición del abecedario en los distintos estímulos, que va reduciéndose tras cada elección. Además, el parpadeo y presentación de la selección realizada por el usuario por pantalla también deben tratarse desde distintos puntos de vista, dado que en el caso del deletreo deberá presentarse la palabra que se va formando, mientras que en la comunicación mediante ideogramas el *feedback* al usuario depende de qué tipo de elección se esté realizando (manifestación de una necesidad o selección concreta de una solución a la misma).

Por otro lado, la **integración con el sistema** completo tiene un papel crucial en el proyecto, ya que la mayor parte de la validación y experimentación depende de las etapas de captura y análisis de la señal. Está se ha realizado mediante una comunicación entre los procesos de análisis y la interfaz gráfica mediante cola de mensajes, para lo cual ha sido necesaria la creación y manipulación de un hilo de ejecución en el proceso de la interfaz que se mantenga a la espera de la información enviada por la etapa de análisis de la señal, y se encargue de la identificación del estímulo seleccionado por el usuario basándose en el dato recibido.

Con respecto a las **pruebas** realizadas, en primer lugar se realizó una prueba de calibración de los estímulos con la ayuda de un fotodiodo, que permitió medir la frecuencia real de parpadeo de los estímulos presentados por pantalla, en la que se pudo empezar a advertir que la precisión de la frecuencia de estimulación mediante el parpadeo de imágenes resultaba mucho más precisa que la alcanzada mediante cuadros de color. En las pruebas finales una vez realizada la integración con el resto de etapas, se confirmó dicha suposición comparando los resultados obtenidos con las dos versiones de la funcionalidad

de deletreo implementadas. Mientras que en los casos de la estimulación mediante el parpadeo de imágenes alcanzaba buenas tasas de acierto por cada selección, en su mayoría, superiores al 50%, incluso casi alcanzando en ciertos casos en torno a un 70%, en el caso de los cuadros de color, muchas ni siquiera alcanzaban dicho porcentaje. Dichas pruebas fueron realizadas con una serie de frecuencias obtenidas previamente mediante el algoritmo de **búsqueda de frecuencias mejor adaptadas al usuario**, consistente en un barrido de un rango de frecuencias y un posterior estudio de la compatibilidad entre ellas. Los resultados obtenidos han sido satisfactorios, ya que las pruebas muestran, en general, una significativa mejoría entre la experimentación con frecuencias prefijadas y las optimizadas para el usuario.

Los resultados de las pruebas realizadas pueden resultar llamativos debido a su gran disparidad en los porcentajes obtenidos, no obstante esto encuentra fácil explicación en que no todos los usuarios se adaptan de igual este tipo de estimulación, dado que hay personas que no son capaces de alcanzar un control apropiado sobre una interfaz cerebro-máquina.

Por otro lado, se advirtió que la limitación de la precisión en la adquisición de señales del casco influía negativamente en la posibilidad de realizar el deletreo, dado que la tasa de error obtenida en una selección es acumulativa debido a la necesidad de realizar tres selecciones hasta llegar a la letra deseada. Esto complica en gran medida que el usuario logre alcanzar finalmente la letra que desea escoger. Sin embargo, este inconveniente puede ser fácilmente solventado utilizando la interfaz implementada haciendo uso de cascos EEG de mayor precisión.

Cabe destacar, que una de las ventajas de esta interfaz cerebro-máquina reside en que realizar un barrido de frecuencias con el usuario una vez es suficiente para controlar las aplicaciones implementadas sin necesidad de volver a entrenar durante un largo periodo de tiempo, dado que los análisis y resultados de cada usuario quedan almacenados. Por otro lado, si la adaptabilidad de las frecuencias al usuario se viera comprometida significativamente a lo largo del tiempo, en cualquier momento podrían volver a realizarse de nuevo fácil y rápidamente la extracción de frecuencias óptimas.

El **resultado final** de este proyecto es muy positivo ya que, pese a la utilización de un caso de bajo coste y a sus limitaciones en la calidad de la adquisición de la señal, se ha conseguido desarrollar una interfaz fácilmente configurable y adaptada al usuario que aporta buenos resultados y puede emplearse en aplicaciones que permiten solventar problemas reales. Además el trabajo realizado servirá como base para futuras investigaciones en el Grupo de Neurocomputación Biológica de la Universidad Autónoma de Madrid.

6.2 Trabajo futuro

A continuación, se exponen algunas posibles líneas de trabajo futuro y mejora de la aplicación de esta interfaz cerebro-maquina.

6.2.1 Tratamiento de la fase del estímulo

Como se comentaba en la sección 3.1, los resultados más precisos y eficientes para interfaces cerebro-máquina hasta la fecha se han conseguido mediante la estimulación visual y el análisis de las respuestas SSVEP a través tanto de la frecuencia como de la fase del estímulo objetivo (Chen et al., 2015).

Para ello, no se realiza una estimulación ininterrumpida de todas las frecuencias, tal y como ha sido implementada en este proyecto, sino que se realiza una estimulación en

intervalos de 0,5 segundos, intercaladas entre intervalos de la misma duración en los que únicamente se presentará la pantalla en negro, carente de estimulación. Todos los periodos de estimulación de cada cuadro presentado por pantalla tendrán la misma frecuencia ($f = 1/T$) y la misma fase (φ), por lo que podrá reconocerse más rápidamente el estímulo en el que el usuario centra su atención visual.

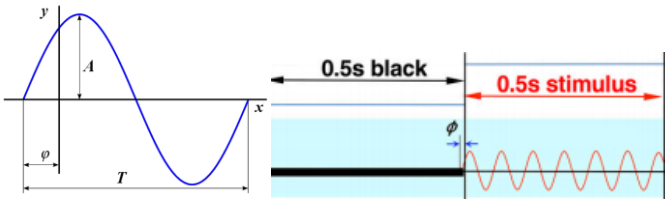


Figura 6-1: Frecuencia y fase de una señal sinusoidal

Esta mejora podría incluirse en la aplicación de la interfaz gráfica sin necesidad de realizar demasiadas alteraciones en el código ya desarrollado, implementándose un nuevo método de control, similar al parpadeo de los estímulos y complementario a este, haciendo uso de los recursos de *signal* y *slot* proporcionados por la librería Qt, empleada en este proyecto, gracias a los cuales, se eliminaría de la pantalla cualquier tipo de estímulo (pantalla en negro) cada y durante un lapso de tiempo determinado previamente. Además, en el hilo de escucha de información proveniente del proceso de análisis de la señal, deberá realizarse la comprobación tanto de la frecuencia como de la fase de la señal recibidas para la identificación del estímulo seleccionado.

6.2.2 Texto predictivo

Otra posible mejora a la aplicación desarrollada podría consistir en la incorporación de una función de predicción de texto, similar a la que implementan los dispositivos móviles en su funcionalidad de teclado, para la funcionalidad de comunicación mediante deletreo, presentada como un estímulo por pantalla. Esto permitiría agilizar en gran medida este tipo de comunicación, dado que se reduciría considerablemente el número de selecciones a realizar, teniendo en cuenta que se necesitan tres por cada letra, puesto que la predicción de la palabra buscada aparecerá por pantalla antes de finalizar el deletreo.

Esto podría llevarse a cabo mediante el uso de diccionarios de palabras más comúnmente utilizadas por los usuarios, realizando una predicción de texto con ayuda de árboles de decisión teniendo en cuenta los caracteres seleccionados por el usuario.

6.2.3 Implementación de memorias de estados anteriores

En la funcionalidad de comunicación por deletreo de palabras, es muy probable que el usuario desee mantener diversas conversaciones y expresar diversas ideas. Por ello, una posible mejora, podría ser añadir "memoria" a la aplicación. De esta manera, el usuario podría elegir entre varios estados previos, presentados en forma de estímulos, y una vez escogido uno, podría retomarse el deletreo a partir de donde el usuario lo dejara.

La memoria de los estados debe perdurar incluso aunque se cierre la aplicación, por lo que una solución a este problema podría consistir en guardar cada deletreo en un archivo adicional de configuración que se leyera al inicio de la ejecución. Dado que la cantidad de estímulos presentada por pantalla es limitada, la aplicación cargaría los últimos estados almacenados, de manera que si se dispusiesen de más estados que estímulos, los más antiguos no formarían parte de las opciones mostradas al usuario.

6.2.4 Otros cascos

Este proyecto ha centrado sus pruebas en el uso del casco EEG de bajo costo Emotiv Epoc, disponible en el laboratorio del Grupo de Neurocomputación Biológica de la Universidad Autónoma de Madrid. Sin embargo, una línea futura bastante evidente sería la adaptación e integración de la aplicación a otros cascos EEG distintos, haciéndola más universal y flexible.

Esta línea de trabajo ya está en proceso para un gorro EEG no inalámbrico de g.tec, g.GAMMAcap perforado siguiendo el sistema 10-20, que utiliza electrodos secos g.SAHARA y el amplificador g.USBamp, disponibles recientemente en el laboratorio del Grupo de Neurocomputación Biológica de la Universidad Autónoma de Madrid.

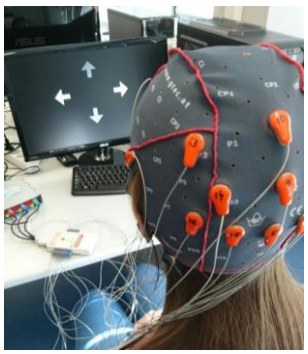


Figura 6-2: Trabajo futuro: adaptación de la interfaz al gorro EEG g.GAMMAcap con electrodos g.SAHARA dry y aplificador g.USBamp

Bibliografía

Abdulkader, S. N., Atia, A., and Mostafa, M.-S. M. (2015). Brain computer interfacing: Applications and challenges. *Egypt. Informatics J.* doi:10.1016/j.eij.2015.06.002.

Baba, A., and Burke, M. (2008). Electrical characterisation of dry electrodes for ECG recording. *WSEAS Int. Conf.*, 76–81. Available at: http://www.wseas.us/e-library/conferences/2008/crete/circuits/circuits10.pdf.

Birbaumer, N. (1999). slow Cortical Potentials: Plasticity, Operant Control, and Behavioral Effects. *Neurosci.* 5, 74–78. doi:10.1177/107385849900500211.

Chang, M. H., Lee, J. S., Heo, J., and Park, K. S. (2016). Eliciting dual-frequency SSVEP using a hybrid SSVEP-P300 BCI. *J. Neurosci. Methods* 258, 104–113. doi:10.1016/j.jneumeth.2015.11.001.

Chen, X., Wang, Y., Nakanishi, M., Gao, X., Jung, T.-P., and Gao, S. (2015). High-speed spelling with a noninvasive brain–computer interface. *Proc. Natl. Acad. Sci.* 112, 1–10. doi:10.1073/pnas.1508080112.

Cipresso, P., Carelli, L., Solca, F., Meazzi, D., Meriggi, P., Poletti, B., et al. (2012). The use of P300-based BCIs in amyotrophic lateral sclerosis: from augmentative and alternative communication to cognitive assessment. *Brain Behav.* 2, 479–98. doi:10.1002/brb3.57.

Cui, J., Wong, W., and Mann, S. (2004). Time-frequency analysis of visual evoked potentials by means of matching pursuit with chirplet atoms. *Conf. Proc. ... Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. IEEE Eng. Med. Biol. Soc. Annu. Conf.* 1, 267–70. doi:10.1109/IEMBS.2004.1403143.

Dix, A. (2009). "Human-computer interaction," in *Encyclopedia of Database Systems*, ed. Springer US, 1327–1331.

Duszyk, A., Bierzyńska, M., Radzikowska, Z., Milanowski, P., Kuś, R., Suffczyński, P., et al. (2014). Towards an optimization of stimulus parameters for brain-computer interfaces based on steady state visual evoked potentials. *PLoS One* 9, e112099. doi:10.1371/journal.pone.0112099.

Emotiv (2016). Emotiv EPOC / EPOC+. Available at: https://emotiv.com/epoc.php.

Fernandez-Vargas, J., Pfaff, H. U., Rodríguez, F. B., and Varona, P. (2013). Assisted closed-loop optimization of SSVEP-BCI efficiency. *Front. Neural Circuits* 7, 27. doi:10.3389/fncir.2013.00027.

g.tec Medical Engineering (2015). g.Software - Software options. Available at: http://www.gtec.at/Products/Software/Software-options.

g.tec Medical Engineering (2016). g.Nautilus - Wireless Biosignal Aquisition. Available at: http://www.gtec.at/Products/Hardware-and-Accessories/g.Nautilus-Specs-Features.

Galindo Merchán, D. (2008). Introducción a los sistemas Brain Computer Interface : La Cofa – Blog de los empleados de Telefónica I+D. Available at: http://www.lacofa.es/blog/2008/12/15/introduccion-a-los-sistemas-brain-computer-interface/.

Gembler, F., Stawicki, P., and Volosyak, I. (2015). Autonomous Parameter Adjustment for SSVEP-Based BCIs with a Novel BCI Wizard. *Front. Neurosci.* Available at: http://dx.doi.org/10.3389/fnins.2015.00474.

Graimann, B., Allison, B., and Pfurtscheller, G. (2010). "Brain-Computer Interfaces - Revolutionizing Human-Computer Interaction," in *Brain-Computer Interfaces*, 1–27. doi:10.1007/978-3-642-02091-9.

Herrmann, C. S. (2001). Human EEG responses to 1-100 Hz flicker: Resonance phenomena in visual cortex and their potential correlation to cognitive phenomena. *Exp. Brain Res.* 137, 346–353. doi:10.1007/s002210100682.

Hornero, R., Corralejo, R., and Álvarez, D. (2012). Brain-Computer Interface (BCI) aplicado al entrenamiento cognitivo y control domótico para prevenir los efectos del envejecimiento. *Lychnos*. Available at: http://www.fgcsic.es/lychnos/es_es/articulos/Brain-Computer-Interface-aplicado-al-entrenamiento-cognitivo.

HTML Color Codes (2016). Códigos de colores HTML. *HTML Color Codes*. Available at: http://html-color-codes.info/codigos-de-colores-hexadecimales/.

Kuś, R., Duszyk, A., Milanowski, P., Łabęcki, M., Bierzyńska, M., Radzikowska, Z., et al. (2013). On the Quantification of SSVEP Frequency Responses in Human EEG in Realistic BCI Conditions. *PLoS One* 8, e77536. doi:10.1371/journal.pone.0077536.

Lebedev, M. (2014). Brain-machine interfaces: an overview. *Transl. Neurosci.* 5, 99–110. doi:10.2478/s13380-014-0212-z.

Leuthardt, E. C., Miller, K. J., Schalk, G., Rao, R. P. N., and Ojemann, J. G. (2006). Electrocorticography-based brain computer interface - The seattle experience. in *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 194–198. doi:10.1109/TNSRE.2006.875536.

Monge, S. (2015). fMRI – Resonancia Magnética Funcional - Neuromarca. *Neuromarca*. Available at: http://neuromarca.com/neuromarketing/fmri.

Mora-Cortes, A., Manyakov, N., Chumerin, N., and Van Hulle, M. (2014). Language Model Applications to Spelling with Brain-Computer Interfaces. *Sensors* 14, 5967–5993. doi:10.3390/s140405967.

Morán García, Á. (2015). Diseño de interfaces cerebro-máquina controlados mediante registros de EEG. Available at: http://arantxa.ii.uam.es/~jms/pfcsteleco/lecturas/20150203AlvaroMoranGarcia.pdf.

Mulder, T. (2007). Motor imagery and action observation: Cognitive tools for rehabilitation. *J. Neural Transm.* 114, 1265–1278. doi:10.1007/s00702-007-0763-z.

Neuper, C., W??rtz, M., and Pfurtscheller, G. (2006). Chapter 14 ERD/ERS patterns reflecting sensorimotor activation and deactivation. *Prog. Brain Res.* 159, 211–222. doi:10.1016/S0079-6123(06)59014-4.

Nicolas-Alonso, L. F., and Gomez-Gil, J. (2012). Brain computer interfaces, a review. *Sensors* 12, 1211–1279. doi:10.3390/s120201211.

OpenBCI (2016). OpenBCI - Downloads. Available at: http://openbci.com/index.php/downloads.

OpenViBE (2016). OpenViBE - Software for Brain Computer Interfaces and Real Time Neurosciences. Available at: http://openvibe.inria.fr/downloads/.

Prieto, A., Pelayo, F., López, M. Á., and Romero, S. (2013). Brain Computer Intefaces (BCI). *Cent. Mediterráneo la UGR*. Available at: http://atc.ugr.es/pages/docencia/no_reglada/tendencias_ic/media/_doc/bci_aprieto_julio2013_v1/!

Pultarova, T. (2014). Brain-computer interface project gives hope to disabled. *Eng. Technol. Mag.*

Resalat, S. N., Saba, V., Afdideh, F., and Heidarnejad, A. (2012). High-speed SSVEP-based BCI: Study of various frequency pairs and inter-sources distances. in *Proceedings - IEEE-EMBS International Conference on Biomedical and Health Informatics: Global Grand Challenge of Health Informatics, BHI 2012*, 220–223. doi:10.1109/BHI.2012.6211550.

Singla, R., Khosla, A., and Jha, R. (2013). Influence of stimuli color on steady-state visual evoked potentials based BCI wheelchair control. *J. Biomed. Sci. Eng.* 06, 1050–1055. doi:10.4236/jbise.2013.611131.

Swartz Center for Computational Neuroscience (2016). EEGLAB. Available at: http://sccn.ucsd.edu/eeglab/.

The Qt Company (2016). Qt. Available at: http://www.qt.io/.

Trans Cranial Technologies (2012). 10/20 System Positioning Manual. 20. Available at: https://www.trans-cranial.com/local/manuals/10_20_pos_man_v1_0_pdf.pdf.

Yin, E., Zhou, Z., Jiang, J., Yu, Y., and Hu, D. (2015). A dynamically optimized SSVEP brain-computer interface (BCI) speller. *IEEE Trans. Biomed. Eng.* 62, 1447–1456. doi:10.1109/TBME.2014.2320948.

Zhu, D., Bieger, J., Garcia Molina, G., and Aarts, R. M. (2010). A survey of stimulation methods used in SSVEP-based BCIs. *Comput. Intell. Neurosci.* 2010, 702357. doi:10.1155/2010/702357.

Glosario

Armónicos Múltiplo de la frecuencia fundamental. El primer armónico de una frecuencia es el doble del valor de la misma.

Baseline Línea Basal. Estado de reposo del individuo o registro de la actividad cerebral sin ser generada por ningún tipo de estimulación visual.

BCI *Brain-Computer Interface*. Interfaz Cerebro-Ordenador.

BMI *Brain-Machine Interface*. Interfaz Cerebro-Máquina.

C++ Lenguaje de programación orientada a objetos surgido a partir del lenguaje de programación estructurada C.

EEG Electroencefalograma/Electroencefalografía. Exploración neurofisiológica que se basa en el registro de la actividad bioeléctrica del cerebro.

ELA Esclerosis Lateral Amiotrófica. Enfermedad degenerativa neuromuscular. Produce parálisis progresiva muscular.

Electrodo Extremo de un conductor que hace contacto con un medio no metálico. En el caso de los electrodos del casco EEG suponen el contacto entre el circuito y la cabeza del usuario.

Enventanado de la señal Limitación en el tiempo de una señal continua para su análisis.

ERD *Event-Related Desychronization*. Atenuación de la amplitud de los ritmos sensoriomotores

ERS *Event-Related Synchronization* Aumento de la amplitud de los ritmos sensoriomotores

Estacionaridad Invariación de un sistema lo largo del tiempo.

Estímulo Señal externa o interna capaz de provocar una reacción en un organismo. El estímulo visual es un estímulo sensorial percibido por el sentido de la vista.

Feedback Retroalimentación. Implica que cierta información de salida del sistema sea conducida a su entrada.

FFT - DFT Función de Transformada de Fourier. Transformación matemática de las señales entre el dominio del tiempo y el dominio de la frecuencia.

GNB Grupo de Neurocomputación Biológica de la Universidad Autónoma de Madrid

HMI Human Machine Interface. Interfaz Hombre-Máquina

LED Diodo emisor de luz.

Occipital Zona situada en la región posterior del cráneo.

Qt Librería de programación de interfaces gráficas orientada a la optimización de tiempos. Implementada en el lenguaje de programación C++.

SDK *Software Development Kit*. Conjunto de herramientas de desarrollo software que permite crear aplicaciones para un sistema determinado.

Solapamiento de ventanas En el enventanado de señales, número de muestras que una ventana y la anterior comparten.

SSVEP *Steady State Visual Evoked Potentials*. Señales naturales generadas en respuesta a una estimulación visual a unas frecuencias determinadas. Dichos potenciales tienen una frecuencia muy similar a la estimulada.

Subarmónicos Divisores de la frecuencia fundamental.

Timeout Señal generada como una alarma cuando el *timer* alcanza un valor determinado.

Timer Función que permite el manejo del paso del tiempo

Anexos

A Manual de instalación

Para hacer uso de la interfaz cerebro-máquina diseñada y desarrollada en este proyecto, es necesaria la instalación de una serie de librerías relacionadas con el software del casco EEG que se va a utilizar, así como las librerías en las que se basa la interfaz gráfica.

- **Librerías para la gestión del casco EEG**: La SDK del casco Emotiv Epoc proporciona dichas librerías, permitiendo el acceso a los datos brutos tomados por el casco. En este caso son necesarias *ledk*, *ledk_utils_linux*, y las librerías que éstas necesitan para funcionar, que se encuentran en la subcarpeta *lib*, dentro de la carpeta de instalación de la SDK, en su última versión 2.0.0. Además es compatible con distintas versiones para la compilación.

- **Librería para la creación de la interfaz gráfica (Qt)**: Es necesaria la librería Qt de diseño de interfaces gráficas en C++, en su última versión 5.5.1:

```
wget http://download.qt.io/official_releases/qt/5.5/5.5.1/qt-opensource-linux-x64-5.5.1.run

chmod +x qt-opensource-linux-x64-5.5.1.run

./ qt-opensource-linux-x64-5.5.1.run
```

  Para poder emplearla se deberán instalar una serie de librerías necesarias para el correcto funcionamiento de la principal, como un compilador de C++ y unas librerías OpenGL:

```
sudo apt-get install build-essential

sudo apt-get install mesa-common-dev

sudo apt-get install libglu1-mesa-dev-y
```

- **Librerías para realizar operaciones matemáticas GSL**: Para ciertas operaciones utilizadas en el enventanado de la señal, son necesarias las librerías *GNU Scientific Library*. El método más rápido para la instalación de estas librerías es mediante consola a través del *Advanced Packaging Tool*:

```
sudo apt-get install libgsl0dbl

sudo apt-get install gsl-bin libgsl0-dev

sudo apt-get install gsl-doc-info gsl-doc-pdf gsl-ref-html gsl-ref-psdoc
```

B Manual del programador

En este anexo se documentan las clases creadas para la implementación de la interfaz gráfica desarrollada en este proyecto, destinada al uso de una interfaz cerebro-máquina ofreciendo las funcionalidades de desplazamiento y comunicación mediante deletreo y presentación por pantalla de ideogramas.

```
#ifndef MYTHREAD_H
#define MYTHREAD_H

#include <QThread>
#include "mywindow.h"

#include <syscall.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <sys/msg.h>


// Constantes para la creación de la cola de mensajes
#define IPC_DATOS 1
#define IPC_FIN 2

/**
  @brief Estructura de datos para la comunicación entre el proceso de análisis y la interfaz
  tipo de comunicación - IPC_FIN en caso de que se produzca el fin del establecimiento de la conexión o IPC_DATOS en cualquier otro caso.
  Comando - dato con valor de la frecuencia analizada
*/
typedef struct
{
    long tipoComunicacion;
    int Comando;

}Mensaje;


class MyThread : public QThread
{
    Q_OBJECT
protected:
    /**
      @brief Override del método run de la clase QThread.
      Se realiza la recepción de datos del proceso de análisis de la señal
      y la comparación de los mismo con las frecuencias estimuladas
    */
    void run();

};

#endif // MYTHREAD_H
```

Figura MP-1: Clase MyThread

```
#ifndef MYWINDOW_H
#define MYWINDOW_H

#include <QWidget>
#include <QVBoxLayout>
#include <QTimer>
#include <QPlainTextEdit>
#include <QMainWindow>
#include <QCoreApplication>
#include <QMutex>
#include <string.h>
#include "stimulus.h"
#include "mythread.h"

#define MAX_STIM 8

class Mywindow : public QWidget
{
    Q_OBJECT

private:
    static bool instanceFlag;
    static Mywindow * myinstance;


public:
    Stimulus * stimulus[MAX_STIM];  /*Estímulos de la ventana principal*/
    int num_stimulus[MAX_APP];      /*Número de estímulos de la ventana*/
    QVBoxLayout *layout[MAX_STIM];  /*Layout para colocar los estímulos*/
    QTimer *timer;                  /*Variable para el control del tiempo*/
    char abec[4][100];              /*Abecedario para la primera selección del deletreo*/

public:

    /**
      @brief Constructor de la clase/widget Mywindow para la creación de la ventana principal
      @param parent - Identificador del widget en el que se encuentra Mywindow. Si es 0, el widget creado será una nueva ventana
      @return the result of the operation, being cd_Error_NoError if all worked OK
      */
    Mywindow(QWidget *parent = 0);

    /**
      @brief Método que devuelve la instacia de la clase Mywindow, necesaria para la comunicación con el hilo
      @return Devuelve un puntero al objeto de tipo Mywindow
     */
    static Mywindow* getInstance();

    /**
      @brief Método que establece las características básicas de la ventana principal: título, tamaño y color de fondo
      @param title - cadena de caracteres con la información del título de la aplicación
    */
    void setWindow(const char *title);

    /**
      @brief Método que establece el número de estímulos presentados por pantalla como opciones al usuario
      @param num - número de estímulos presentados por pantalla como opciones al usuario
    */
    void setNumStimulus(int num);


    /**
      @brief Método que configura cada estímulo mediante su posición, tamaño y timer
      @param i - identificador del estímulo que se quiere incluir a la ventana
      @param type - tipo de aplicación al que va a pertener el estímulo que se pretende incluir a la ventana
     */
    void setupStimulus(int i, int type);

    /**
      @brief Método que actualiza los estímulos presentados por pantalla después de cada selección realizada por el usuario.
      @param ind_stim - identificador del estímulo que ha sido seleccionado por usuario. Dato procedente del proceso de análisis de la señal
    */
    void refreshStimulus(int ind_stim);

    /**
      @brief Método que establece el inicio del timer que controla el tiempo de ejecución de la interfaz.
      Una vez se cumple el timeout se ejecuta el slot closeWindow()
      Útil para la grabación del baseline y el entrenamiento de la interfaz para la búsqueda de frecuencias óptimas
      @param time - tiempo en segundos para que se cumpla el timeout
    */
    void setTimeWindow(int time);

    /**
      @brief Método de gestión de creación y envío de eventos al sistema
      Es llamada por el hilo receptor de mensajes procedentes del proceso de análisis de la señal
      Útil para la comunicación con el proceso principal de la interfaz
      @param pos - identificador del estímulo seleccionado por el usuario
    */
    void myEvent(int pos);
```

```
protected:
    /**
      @brief Método para la gestión de recepción de eventos al sistema
      Se encarga de llamar al método para la actualización de estímulos refreshStimulus()
      @param pos - identificador del estímulo seleccionado por el usuario
    */
    void rcvEvent(QKeyEvent *event); /*Método para cerrar la ventana principar si se pulsa ESC*/


public slots:
    /**
      @brief Método que se encarga del cierre de la aplicación
    */
   void closeWindow();
};

#endif // MYWINDOW_H
```

Figura MP-2: Clase Mywindow

```
#ifndef STIMULUS_H
#define STIMULUS_H

#include <QWidget>
#include <QLabel>
#include <QTimer>
#include <math.h>
#include <stdio.h>

/*Constantes para la saber qué tipo de app cargar*/
#define MAX_APP 5
#define INICIO 0
#define MOV 1
#define SPELL 2
#define SPELL2 3
#define IDEO 4

class Stimulus : public QWidget
{
    Q_OBJECT

public:
    int counterini;             /*Contador para modificar el estado del estímulo de incio de encendido a apagado*/
    int counter;                /*Contador para modificar el estado del estímulo de encendido a apagado*/
    QTimer *timer;              /*Variable para el control del tiempo*/

    QLabel *label;              /*Superficie donde se dibujará el estímulo*/
    char * abecedario;          /*Abecedario*/
    int type;                   /*tipo de app*/
    QPixmap frame[MAX_APP];     /*Imagen que representa estímulo encendido*/
    char framesIdeo[4][100];    /*Rutas de imagen que representa estímulo de ideogramas después de la primera elección*/
    char framesAbc[21][100];    /*Rutas de imagen que representa estímulo de deletreo con imagenes después de la primera elección*/

    int identification;         /*ID del estímulo*/
    int height[MAX_APP];        /*Altura del cuadro que representa el estímulo*/
    int width[MAX_APP];         /*Altura del cuadro que representa el estímulo*/
    int posx[MAX_APP];          /*Posición en el eje x (horizontal) del estímulo*/
    int posy[MAX_APP];          /*Posición en el eje y (vertical) del estímulo*/
    float frequency[MAX_APP];   /*Frecuencia asociada al estímulo*/

    int R;                      /*R del fondo del estímulo*/
    int G;                      /*G del fondo del estímulo*/
    int B;                      /*B del fondo del estímulo*/

    bool ind;                   /*Estímulo es un indicador de seleccion*/
    bool train;                 /*Estímulo es un marcador de posición de entrenamiento*/
    bool show;                  /*Flag que indica que el estímulo indicador de seleccion se debe mostrar*/
    bool showTrain;             /*Flag que indica que el estímulo indicador del entrenamiento se debe mostrar*/

public:
    /**
      @brief Constructor de la clase/widget Stimulus para la creación de los estímulos
      @param parent - Identificador del widget en el que se encuentra
      @param id - Identificador del estímulo
    */
    Stimulus(QWidget *parent = 0, int id = 0);
```

```
/**
  @brief Método que crea el estímulo a partir de su posición y su tamaño
  @param posix - Posición del estímulo con respecto al margen izquierdo
  @param posiy - Posición del estímulo con respecto al margen de arriba
  @param sizex - Ancho del estímulo
  @param sizey - Alto del estímulo
*/
void setLabel(int posix, int posiy, int sizex, int sizey);

/**
  @brief Método que establece la imagen del estímulo
  @param image - imagen del estímulo
*/
void setFrame(QPixmap &image);

/**
  @brief Método que establece la parte del abecedario que mostrará dicho estímulo en la funcionalidad de deletreo
  @param abc - cadena de caracteres con parte del abecedario
*/
void setAbecedario(char * abc);


/**
  @brief Método que establece el tipo de funcionalidad/aplicación a la que pertenece el estímulo
  @param tipo - tipo de aplicación
*/
void setType(char typeapp);

/**
  @brief Método que convierte la frecuencia de parpadeo del estímulo a milisegundos para controlar el timer
  @param parent - Identificador del widget en el que se encuentra
  @param id - Identificador del estímulo
  @return the result of the operation, being cd_Error_NoError if all worked OK
*/
int freqToMilisec(float freq);

/**
  @brief Método que establece la frecuencia a la que parpadea el estímulo
  @param freq - frecuencia de parpadeo
*/
void setFrequency(float freq);

/**
  @brief Método que establece el alto del estímulo
  @param heightstim - alto del estímulo
*/
void setHeigth(int heightstim);

/**
  @brief Método que establece el ancho del estímulo
  @param widthstim - ancho del estímulo
*/
void setWidth(int widthstim);

/**
  @brief Método que establece la posición con respecto al margen izquierdo
  @param posix - posición con respecto al margen izquierdo
*/
void setPosX(int posix);

/**
  @brief Método que establece la posición con respecto al margen de arriba
  @param posiy - posición con respecto al margen de arriba
*/
void setPosY(int posiy);

/**
  @brief Método para establecer la cantidad de rojo del cuadro de color
  @param R - código R del RGB
*/
void setR(int erre);

/**
  @brief Método para establecer la cantidad de verde del cuadro de color
  @param G - código G del RGB
*/
void setG(int ge);

/**
  @brief Método para establecer la cantidad de azul del cuadro de color
  @param B - código B del RGB
*/
void setB(int be);

/**
  @brief Método que establece el inicio del timer que controla su parpadeo.
  Útil para la grabación del baseline y el entrenamiento de la interfaz para la búsqueda de frecuencias óptimas
  @param time - tiempo en segundos para que se cumpla el timeout
*/
void startTimer();
```

```
    /**
     @brief Método que para el timer del estímulo
    */
    void stopTimer();

signals:

public slots:

    /**
     @brief Método encargado del parpadeo de las imágenes y los cuadros de color, alternando entre el fondo y el estímulo
    */
    void blink();

};

#endif // STIMULUS_H
```

Figura MP-3: Clase Stimulus

C Manual de usuario

Una vez se ha realizado la correcta instalación de las librerías necesarias para la ejecución de la interfaz cerebro máquina implementada, podrá iniciarse por consola el programa completo, integrada la interfaz con las etapas de captura y análisis de la señal mediante el siguiente comando:

```
./AnalisisSSVEP_EE
```

Por consola se mostrará el menú principal donde se ofrecerán las distintas opciones:

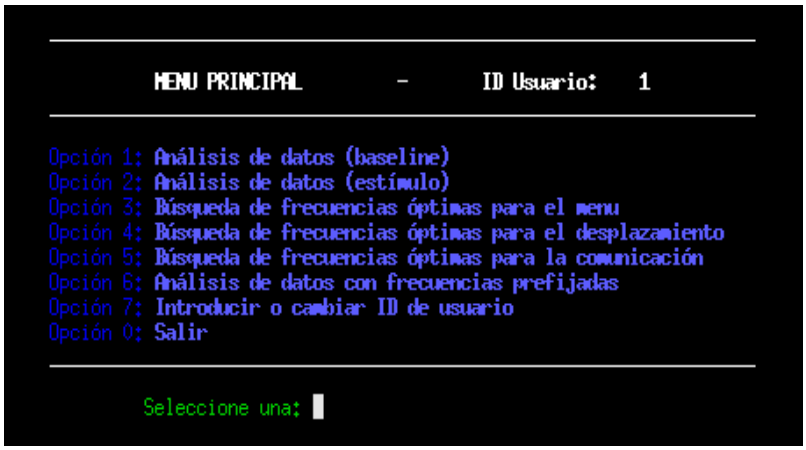


Figura MU-1: Menú principal por consola

Opción 1: Realiza un grabado y análisis de un *baseline* al usuario. Se trata del primero proceso que ha de completarse con un usuario antes de realizar cualquier estimulación. El sujeto deberá mirar a una pantalla en negro durante 60 segundos realizando la menor cantidad de movimientos y parpadeos posibles.

Opción 2: Se realizará una estimulación con las frecuencias obtenidas tras la búsqueda de frecuencias óptimas almacenadas en la carpeta de cada usuario. Se presentará por pantalla el menú principal de selección de funcionalidad mediante estímulos parpadeantes. El proceso que deberá seguir el usuario será especificado más adelante.

Opción 3: Se realizará una búsqueda de frecuencias óptimas para el desplazamiento. Se ofrecerá la opción de entrenar la interfaz con el usuario mediante un algoritmo de ciclo cerrado con cada funcionalidad debido a las distintas características de los estímulos en cada una de ellas, principalmente su posición en la pantalla.

Opción 4: Búsqueda de frecuencias óptimas para la comunicación. Al igual que la opción anterior, se entrenará la interfaz con motivo de encontrar las frecuencias que mejor se adaptan al usuario pero este caso con las funcionalidades de comunicación.

Opción 5: Se realizará un análisis de los datos con estimulación haciendo uso de frecuencias prefijadas.

Opción 6: Permitirá cambiar la identificación del usuario, de manera los datos de cada usuario quedarán almacenados en carpetas distintas denominadas por el identificador de usuario, por lo que se podrán utilizar siempre que se quiera.

A continuación, se presenta una vista previa de las distintas pantallas de la interfaz gráfica presentadas al usuario para cada una de las funcionalidades implementadas.

El menú principal de selección de funcionalidad se verá representado mediante tres estímulos centrados en la pantalla como se muestra en Figura MU-2:

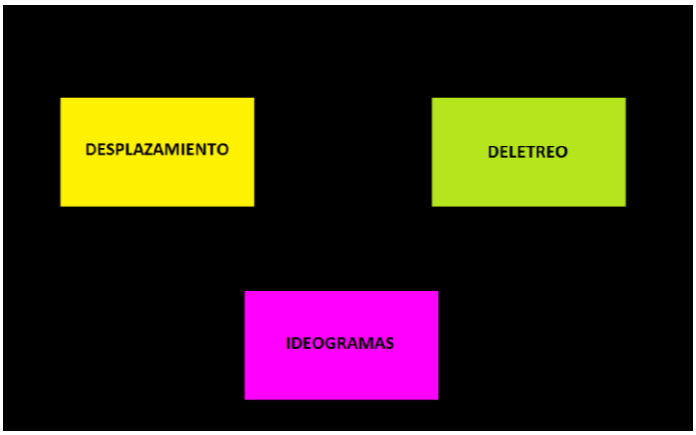


Figura MU-2: Menú principal de la interfaz gráfica de selección de funcionalidad

Si el usuario selecciona la funcionalidad de desplazamiento, por pantalla se presentará la siguiente pantalla, ofreciendo la posibilidad de desplazarse en cuatro direcciones: delante, detrás, izquierda y derecha. Una vez que se escoja un estímulo, se remarcará con un cuadro rojo a su alrededor:

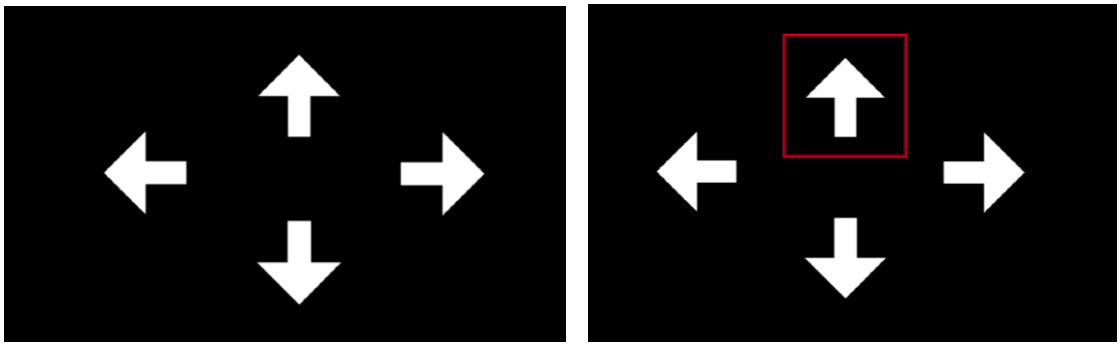
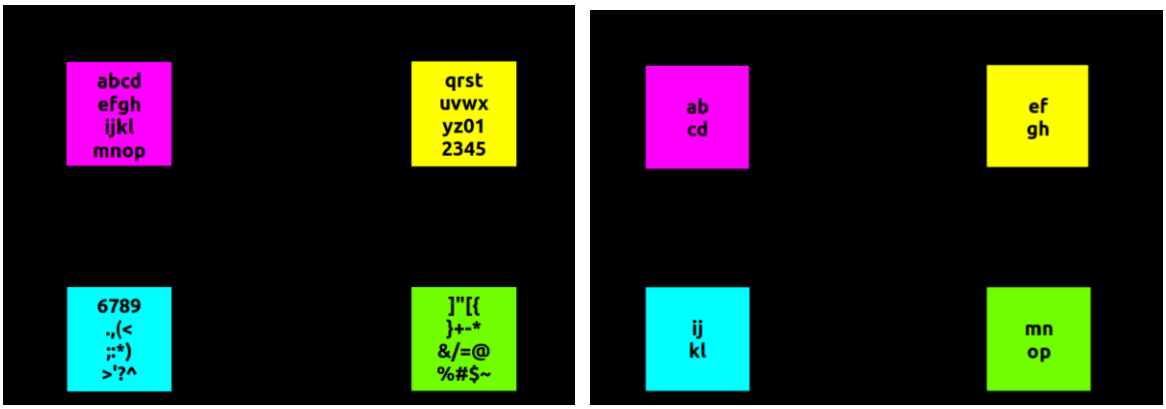


Figura MU-3: Pantallas de funcionalidad de desplazamiento

Si por el contrario, se escoge la funcionalidad de deletreo, se mostrarán cuatro estímulos en los que se verá repartido un conjunto de símbolos y caracteres alfanuméricos mediante los cuales el usuario podrá expresarse deletreando. En la Figura MU-4 puede encontrarse un ejemplo de la sucesión de pantallas para el deletreo de la letra *a*:



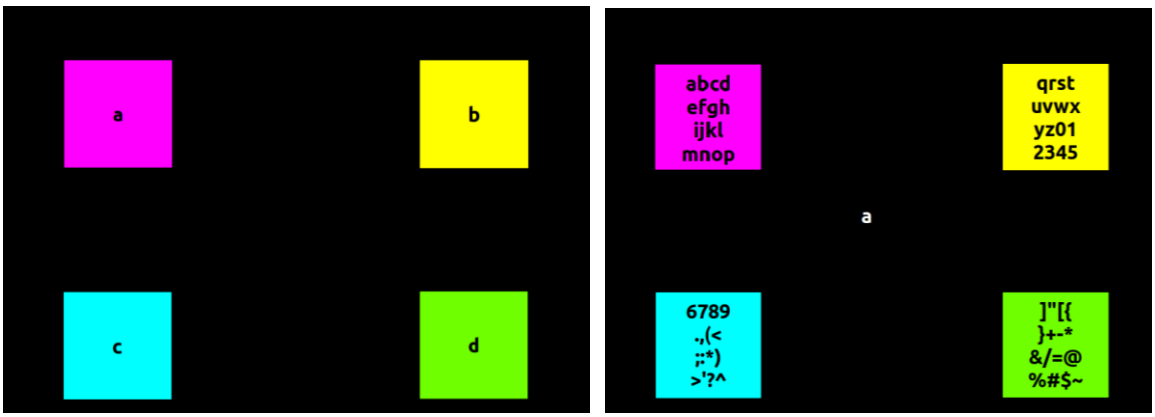


Figura MU-4: Pantallas de funcionalidad de comunicación mediante deletreo

Por último, si se escoge la funcionalidad de comunicación mediante ideogramas, se presentará una primera pantalla ofreciéndole al usuario la oportunidad de elegir la necesidad que desea llevar a cabo entre las cuatro opciones disponibles: comer, beber, descansar y llamada de emergencia. Una vez se haya realizado la primera selección, se ofrecerán cuatro posibilidades para efectuar dicha actividad. La elegida será recuadrada en rojo. En la Figura MU-5 puede observarse un ejemplo concreto de este tipo de comunicación, en el que se selecciona la necesidad de beber un vaso de agua.

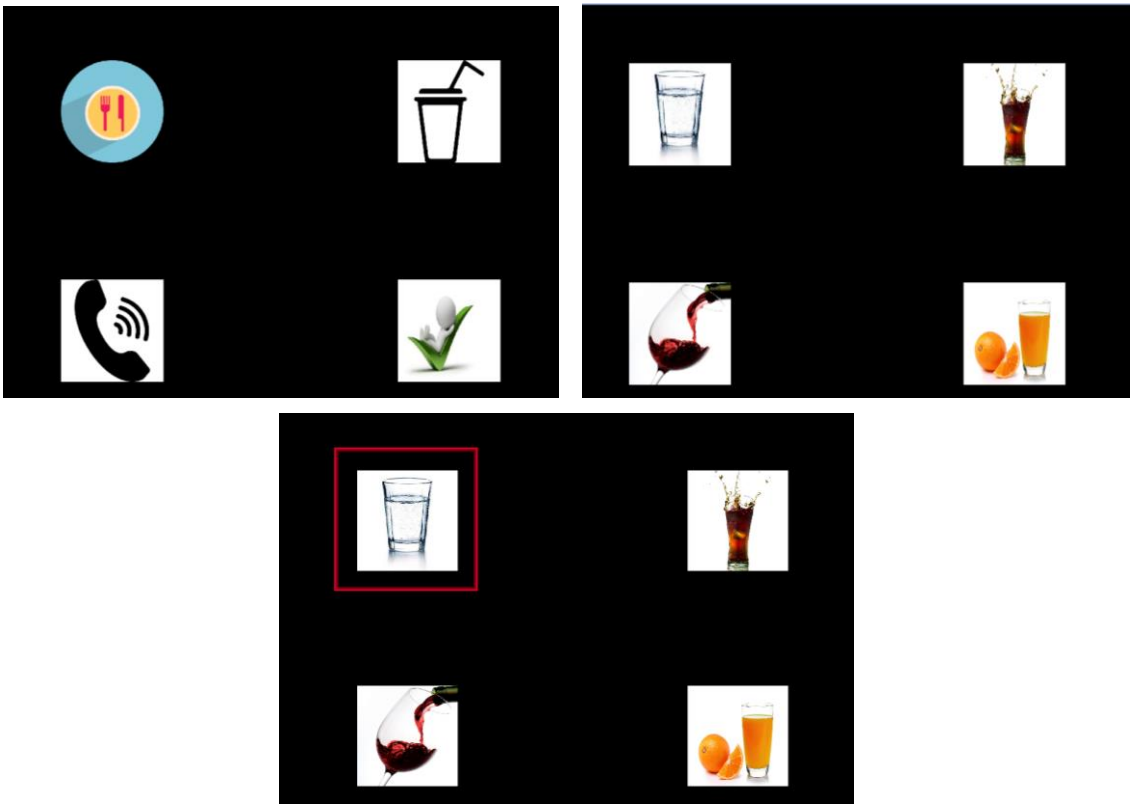


Figura MU-5: Pantallas de funcionalidad de comunicación mediante ideogramas

D Anexo I – Reconocimiento del estímulo seleccionado.

La clase MyThread es la encargada de recibir la frecuencia, obtenida y analizada a partir de las señales cerebrales SSVEP, y realizar la comparación de la misma con cada una de las frecuencias estimuladas. De este modo se podrá reconocer en qué estímulo el usuario ha fijado su atención y se enviará la pulsación de teclado correspondiente a dicho estímulo al proceso principal de la interfaz gráfica.

```
while(1){
    // Lectura del mensaje de la cola
    errorTransmision = msgrcv(idCola, &mensaje, sizeof(mensaje), 0, MSG_NOERROR);
    if (errorTransmision < 0) {
       // printf("    \e[1;31m[ERROR]\e[0m \e[0;37mRecepción de datos fallida\n\e[0m");
        //return;
    }

    if(mensaje.Comando != ultima_freq){
        contador[0] = 0;
        contador[1] = 0;
        contador[2] = 0;
        contador[3] = 0;
    }

    // Analizamos la información según el tipo de dato almacenado en la cola
    switch(mensaje.tipoComunicacion){
    case IPC_FIN:   // Finalizamos la transmisión
    {
        //Llamamos a ESC para cerrar la aplicación
        w->myEvent(99);
        break;
    }

    case IPC_DATOS: // Mensaje típico de datos
    {

        int dato = mensaje.Comando;

        //hacemos la comparación de frecuencias
        for(int i = 0; i < 4; i++){

            if(dato == (int)frecuencias[i]){

                contador[i] = contador[i] + 1;

                if (contador[i] >= 2){ //doble comprobación

                    contador[0] = 0;
                    contador[1] = 0;
                    contador[2] = 0;
                    contador[3] = 0;

                    //pulsacion virtual de tecla
                    w->myEvent(i);
                }

                ultima_freq = frecuencias[i];
            }
        }
        break;
    }
    }// Fin del switch

}
```

Figura AI-1: Código del hilo: Recepción de dato analizado y reconocimiento del estímulo

E Anexo II – Actualización de estímulos

La actualización de los estímulos se lleva a cabo mediante la gestión de los eventos de al sistema, implementada en las funciones *myEvent()*, que se encarga del envío de las pulsaciones virtuales facilitando la comunicación entre los procesos de análisis de señal y la interfaz gráfica, y *recvEvent()*, que se encarga de recibirlos y realizar la llamada a la función *refreshStimulus()*, indicando el identificador correspondiente al estímulo que debe actualizarse.

```
void Mywindow::refreshStimulus(int ind_stim){

    if(stimulus[ind_stim]->type==INICIO){
        /*Se carga la funcionalidad escogida*/

        for(int i = 0; i < num_stimulus[INICIO]; i++){
            if(ind_stim==0){ //MOV

                stimulus[i]->setType(MOV);
                stimulus[num_stimulus[MOV]]->setType(MOV);
                setupStimulus(i, MOV);

            }else if(ind_stim==1){ //SPELL

                stimulus[i]->setType(SPELL2);
                stimulus[num_stimulus[SPELL2]]->setType(SPELL2);
                setupStimulus(i, SPELL2);

            }else if(ind_stim==2){ //IDEO

                stimulus[i]->setType(IDEO);
                stimulus[num_stimulus[IDEO]]->setType(IDEO);
                setupStimulus(i, IDEO);
            }
        }
    }else if(stimulus[ind_stim]->type==MOV){
        /*Se indica con un recuadro el estímulo escogido*/

        stimulus[num_stimulus[MOV]]->setWidth(stimulus[0]->width[MOV]+75);
        stimulus[num_stimulus[MOV]]->setHeigth(stimulus[0]->height[MOV]+75);
        stimulus[num_stimulus[MOV]]->setPosX(stimulus[ind_stim]->posx[MOV]-40);
        stimulus[num_stimulus[MOV]]->setPosY(stimulus[ind_stim]->posy[MOV]-40);
        stimulus[num_stimulus[MOV]]->show=true;

        setupStimulus(num_stimulus[MOV], MOV);

    }else if(stimulus[ind_stim]->type==IDEO){

        if(primeraIdeo == true){
            /*Se cargan las nuevas posibilidades tras la primera elección*/
            for(int i = 0; i < num_stimulus[IDEO]; i++){

                stimulus[i]->setType(IDEO);

                QPixmap imagenueva;
                imagenueva.load(stimulus[ind_stim]->framesIdeo[i]);
                stimulus[i]->setFrame(imagenueva);
                stimulus[num_stimulus[IDEO]]->setType(IDEO);
                setupStimulus(i, IDEO);

            }
            primeraIdeo = false;

        }else{
            /*Se indica con un recuadro el estímulo escogido*/
            stimulus[num_stimulus[IDEO]]->setWidth(stimulus[0]->width[IDEO]+75);
            stimulus[num_stimulus[IDEO]]->setHeigth(stimulus[0]->height[IDEO]+75);
            stimulus[num_stimulus[IDEO]]->setPosX(stimulus[ind_stim]->posx[IDEO]-40);
            stimulus[num_stimulus[IDEO]]->setPosY(stimulus[ind_stim]->posy[IDEO]-40);
            stimulus[num_stimulus[IDEO]]->show=true;
            setupStimulus(num_stimulus[IDEO], IDEO);
        }


    }else if(stimulus[ind_stim]->type==SPELL){
```

```
    int i, j;
    int intervalo = strlen(stimulus[ind_stim]->abecedario)/num_stimulus[SPELL];

    if(intervalo == 0){

        /*Se escribe el texto que estamos construyendo*/
        size_t len1 = strlen(stimulus[num_stimulus[SPELL]]->abecedario);
        size_t len2 = strlen(stimulus[ind_stim]->abecedario);
        char * totalLine = (char*) malloc(len1 + len2 + 1);
        memcpy(totalLine, stimulus[num_stimulus[SPELL]]->abecedario, len1);
        memcpy(totalLine + len1, stimulus[ind_stim]->abecedario, len2);
        totalLine[len1 + len2] = '\0';

        stimulus[num_stimulus[SPELL]]->setType(SPELL);
        stimulus[num_stimulus[SPELL]]->setAbecedario(totalLine);
        stimulus[num_stimulus[SPELL]]->setWidth(300);
        stimulus[num_stimulus[SPELL]]->setHeigth(200);
        stimulus[num_stimulus[SPELL]]->setPosX(825);
        stimulus[num_stimulus[SPELL]]->setPosY(375);
        setupStimulus(num_stimulus[SPELL], SPELL);

        /*Reestablecemos el abecedario*/
        /*Reparticion del abecedario para los estimulos*/
        for(i = 0; i < num_stimulus[SPELL]; i++){
            stimulus[i]->setAbecedario(abec[i]);
            stimulus[i]->setLabel(stimulus[i]->posx[SPELL],stimulus[i]->posy[SPELL], stimulus[i]->width[SPELL], stimulus[i]->height[SPELL]);
        }

    }else{
        char abecedario_origen[strlen(stimulus[ind_stim]->abecedario)];
        strcpy(abecedario_origen, stimulus[ind_stim]->abecedario);
        char abc[num_stimulus[SPELL]][intervalo];

        /*Reparticion del abecedario para los estimulos*/
        for(i = 0; i < num_stimulus[SPELL]; i++){

            for(j = 0;  j < intervalo; j++){
                abc[i][j] = abecedario_origen[j + i * intervalo];
            }

            stimulus[i]->abecedario[0] = '\0';
            abc[i][j] = '\0';

            /*Establecemos los nuevos abecedarios*/
            stimulus[i]->setAbecedario(abc[i]);
            stimulus[i]->setLabel(stimulus[i]->posx[SPELL],stimulus[i]->posy[SPELL], stimulus[i]->width[SPELL], stimulus[i]->height[SPELL]);

        }
    }
}else if(stimulus[ind_stim]->type==SPELL2){

    int i, j;
    int intervalo = strlen(stimulus[ind_stim]->abecedario)/num_stimulus[SPELL2];

    /* Última selección: letra. Establecemos el estímulo indicador con la cadena que se va formando al deletrear*/
    if(intervalo == 0){

        /*Se escribe el texto que estamos construyendo*/
        size_t len1 = strlen(stimulus[num_stimulus[SPELL2]]->abecedario);
        size_t len2 = strlen(stimulus[ind_stim]->abecedario);
        char * totalLine = (char*) malloc(len1 + len2 + 1);
        memcpy(totalLine, stimulus[num_stimulus[SPELL2]]->abecedario, len1);
        memcpy(totalLine + len1, stimulus[ind_stim]->abecedario, len2);
        totalLine[len1 + len2] = '\0';

        /*Vamos dividiendo el texto a la vez que se presentan las imágenes para mostrar finalmente la letra escogida*/
        char abecedario_origen[strlen(stimulus[ind_stim]->abecedario)];
        strcpy(abecedario_origen, stimulus[ind_stim]->abecedario);
        char abc[num_stimulus[SPELL2]][intervalo];

        /*Reparticion del abecedario para los estimulos*/
        for(i = 0; i < num_stimulus[SPELL2]; i++){

            for(j = 0;  j < intervalo; j++){
                abc[i][j] = abecedario_origen[j + i * intervalo];
            }

            stimulus[i]->abecedario[0] = '\0';
            abc[i][j] = '\0';

            /*Establecemos los nuevos abecedarios*/
            stimulus[i]->setAbecedario(abc[i]);
            stimulus[i]->setLabel(stimulus[i]->posx[SPELL2],stimulus[i]->posy[SPELL2], stimulus[i]->width[SPELL2], stimulus[i]->height[SPELL2]);

        }

        if(counterspell2 == 0){

            /*Se cargan las nuevas posibilidades tras la primera elección*/
            for(int i = 0; i < num_stimulus[SPELL2]; i++){

                stimulus[i]->setType(SPELL2);
```

```
                QPixmap imagenueva;
                imagenueva.load(stimulus[i]->framesAbc[ind_stim+1]);
                stimulus[i]->setFrame(imagenueva);
                stimulus[num_stimulus[SPELL2]]->setType(SPELL2);
                setupStimulus(i, SPELL2);

            }
            counterspell2++;
            anterior_stim = ind_stim;

        }else{

            /*Se cargan las nuevas posibilidades tras la segunda elección*/
            for(int i = 0; i < num_stimulus[SPELL2]; i++){

                stimulus[i]->setType(SPELL2);

                QPixmap imagenueva;
                switch(anterior_stim){
                case 0:
                    imagenueva.load(stimulus[i]->framesAbc[anterior_stim+ind_stim+5]);
                    break;
                case 1:
                    imagenueva.load(stimulus[i]->framesAbc[anterior_stim+ind_stim+8]);
                    break;
                case 2:
                    imagenueva.load(stimulus[i]->framesAbc[anterior_stim+ind_stim+11]);
                    break;
                case 3:
                    imagenueva.load(stimulus[i]->framesAbc[anterior_stim+ind_stim+14]);
                    break;
                }
                    imagenueva.load(stimulus[i]->framesAbc[anterior_stim+ind_stim+14]);
                    break;
                }
                stimulus[i]->setFrame(imagenueva);
                stimulus[num_stimulus[SPELL2]]->setType(SPELL2);
                setupStimulus(i, SPELL2);

                counterspell2 = 0;

            }
        }
    }
}
```

Figura AII-1: Código de actualización de estímulo tras una elección